



→ End user

IDE

Chrome

media player.

shell

Hello.c

⇒ Appl<sup>n</sup>  
sw

OS ⇒ Core OS + utility sw + Appl<sup>n</sup> sw  
↓  
Kernel

(control panel, task mgr) (Notepad, paint, ms office, calendar...)

...

CPU

RAM

HDD

Key-board  
(primary) I/O

monitor  
(primary) O/P

etc.

⇒ Hardware  
Resources

① Interface.

② Control prog.

③ Resource Allocation / manager.

End user.

④ Bootable storage device (has contain OS setup)  
CD / DVD / PD / HDD

## \* Functions of Core os / kernel

① Process management

② CPU Scheduling

③ Memory mg<sup>m</sup>

④ file & I/o mg<sup>m</sup>

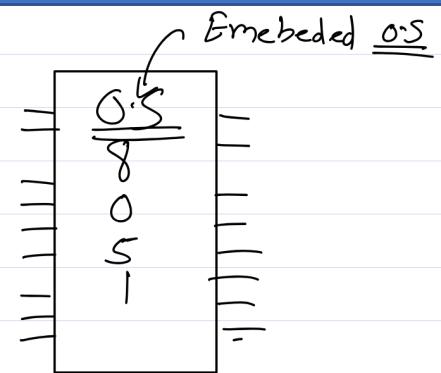
⑤ Hardware Abstraction.

⑥ Networking

⑦ Security and protection

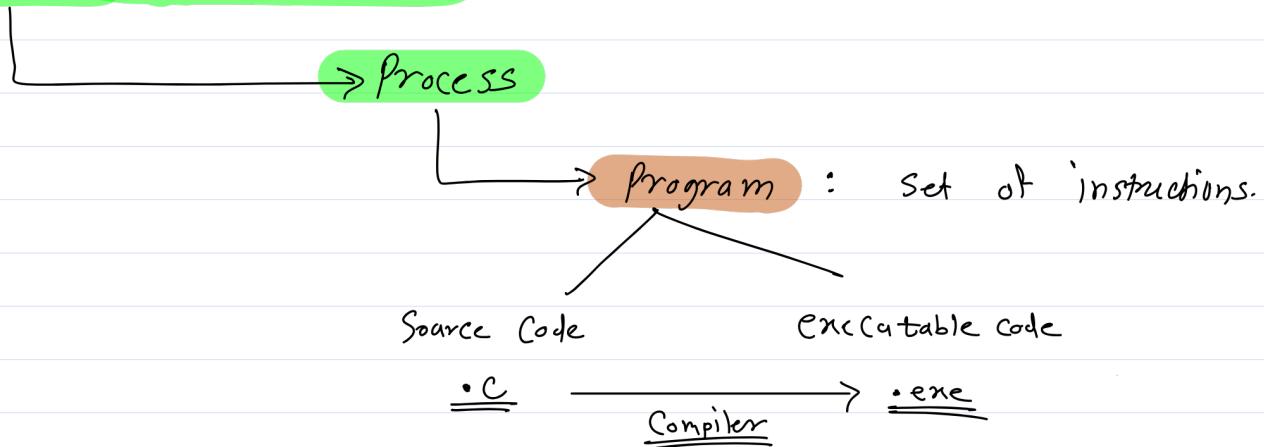
⑧ User interface.

Core Function of OS

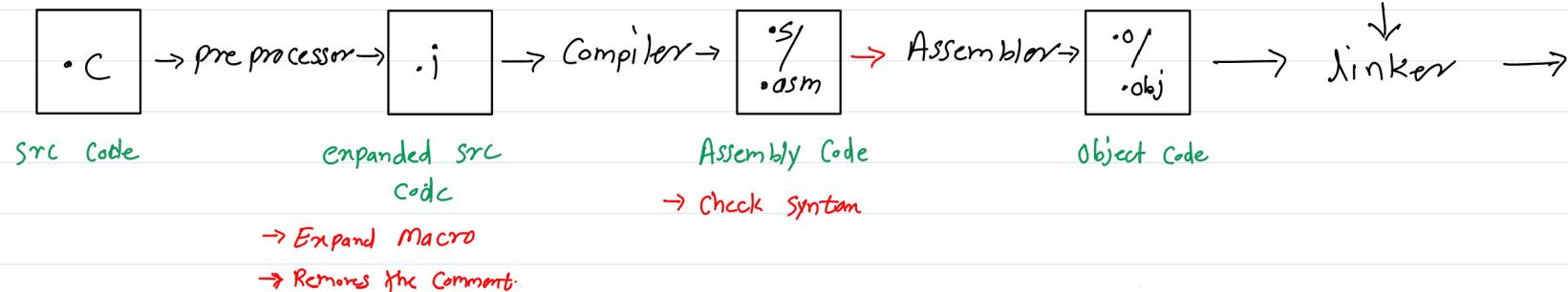


Extra utility fun of OS.

## \* Process Management



## \* Compilation steps:



a.exe  $\Rightarrow$  Windows  
a.out  $\Rightarrow$  Linux/ MAC



(section binary)

• out

exe header or  
primary header

text / code

data

BSS

RO data

Symbol  
table

→ Entry point function address (main)

Information about the Remaining section.

→ Magic Number → uniq Number of file format  
→ Identity of file format.

Char \*P = "Sunbeam"  $\Rightarrow$  RO

\*P = 'x' ; // Error  
putSC P )

machine code

initialised global & static variable

(block started by symbol)

- uninitialised global & static variable

" Constant string literals, 0,1,2, 'A','B' ...

→ The information Related of Variable & Function.

- Name, address, section, size, flags
- These info use for debugging purpose.

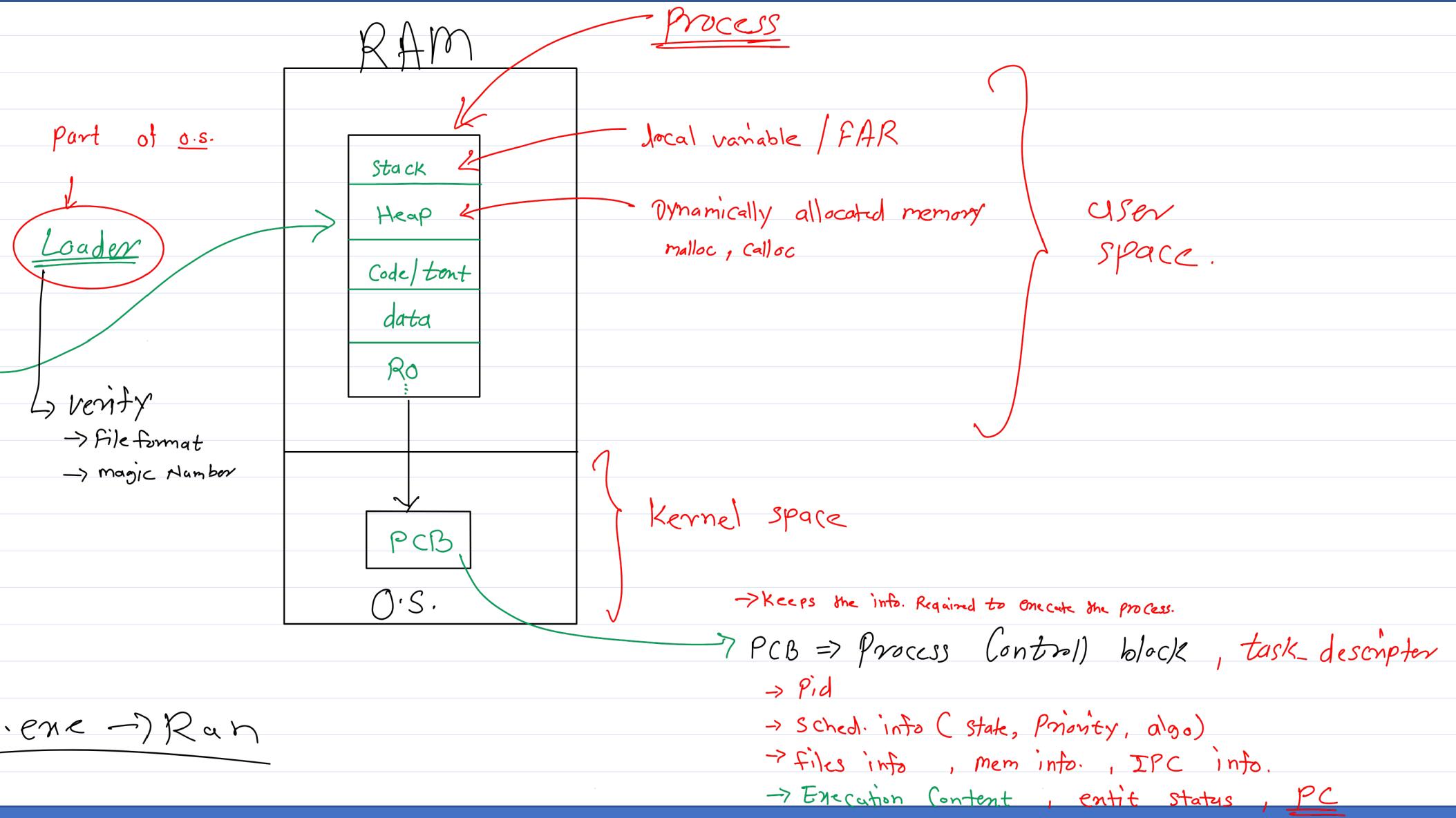
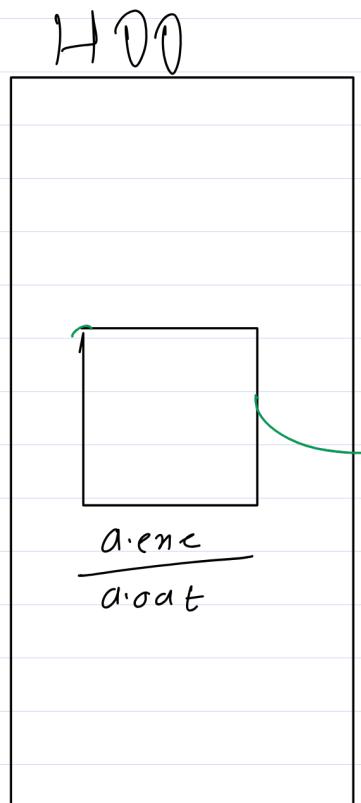
Binary file has magic Number

(It present on starting 2 or 4 bytes of  
of code file)

a • exe  $\rightarrow$  MZ  
a • out  $\rightarrow$  ?ELF  
• bmp  $\rightarrow$   
• pdf  $\rightarrow$   
• text  $\rightarrow$  X

Portable Executable (PE)

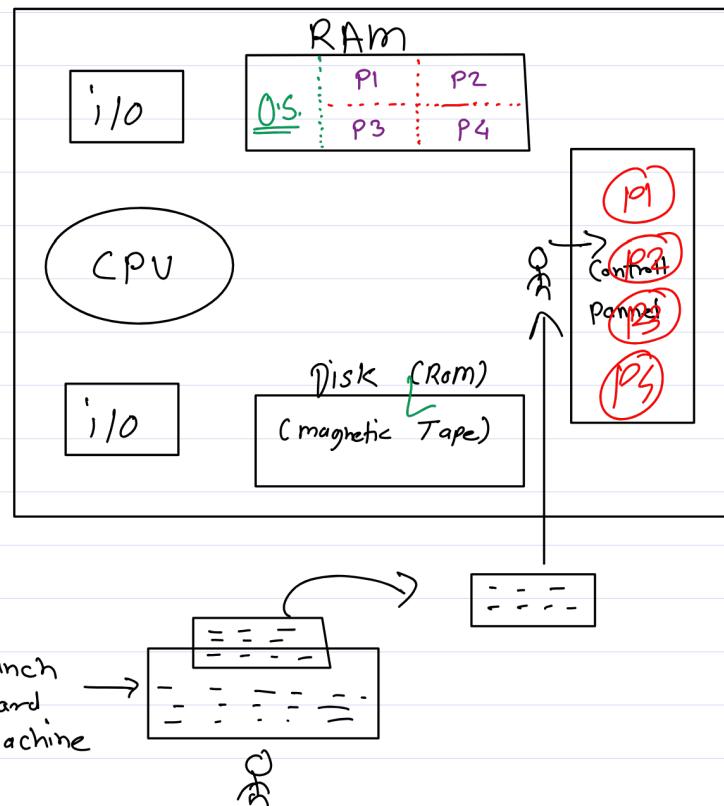
Executable linking format (ELF)



# \* OS Evolution.

## Mainframe Computer

(old computer)



### ① Resident monitor.

### ② Batch System

### ③ Multi-Programming.

- loading multiple programs in main memory.
- Here on RAM load mixed program

↳ CPU bound + I/O bound

- better utilization of CPU

- degree of M.P.

⇒ num of programs that can be kept in RAM.

$sf(n_1)$   
 $sf(n_2)$ 
} I/O instr

$add = n_1 + n_2$   
 $sub = n_1 - n_2$   
 $mul = n_1 * n_2$ 
} CPU instr

$pf(add)$   
 $pf(sub)$   
 $pf(mul)$ 
} I/O instr

$$\text{Program} = \frac{\text{CPU instr}}{\text{CPU burst Time}} + \frac{\text{I/O instr}}{\text{I/O burst time}}$$

$$\left( \frac{\text{CPU burst Time}}{\text{CPU burst Time}} \right) + \left( \frac{\text{I/O burst time}}{\text{I/O burst time}} \right)$$

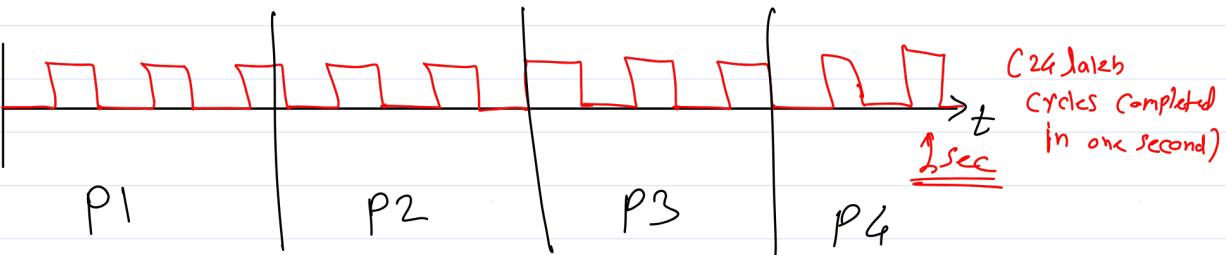
## ④ Time sharing / multitasking O.S.

(2.4GHz)

\* freq : Num of Cycles completed in one sec.

⇒ Sharing CPU time among multiple tasks present in main memory & Ready for execution.

⇒ Response time < 1 sec



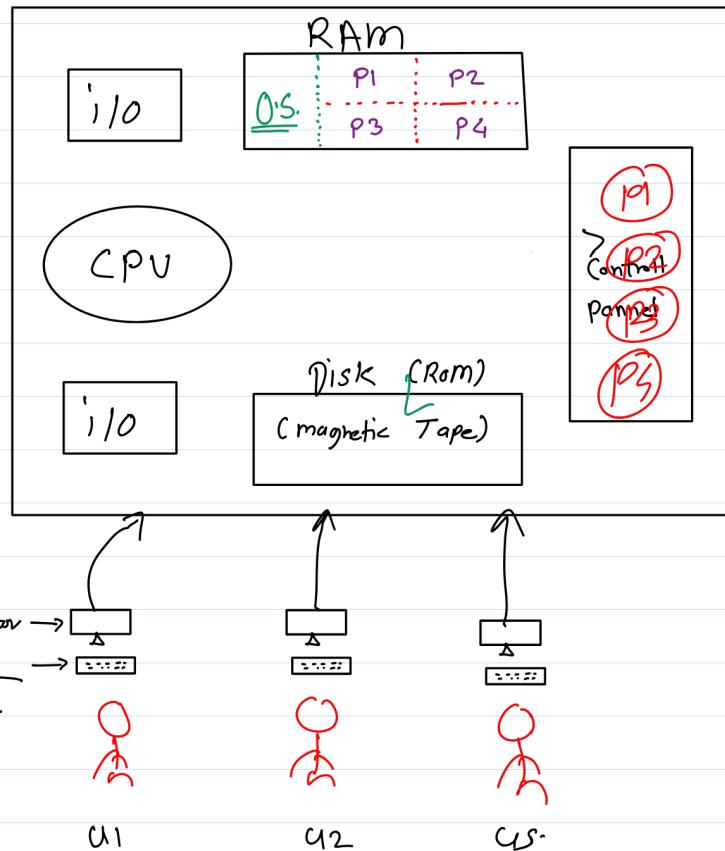
multi - tasking

① Process based  
M.T

② Thread based  
M.T

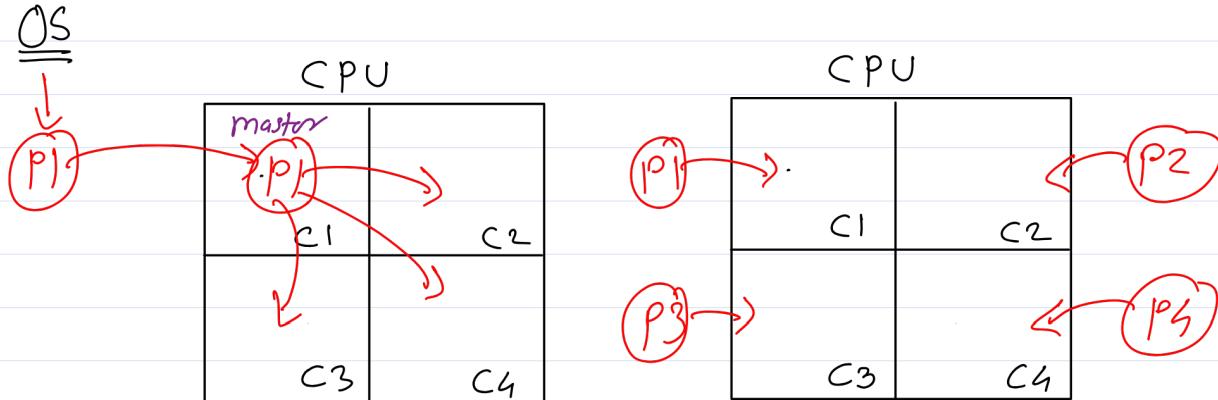
## \* Multi-user System.

puter)



## \* Multi-core / multi-processor.

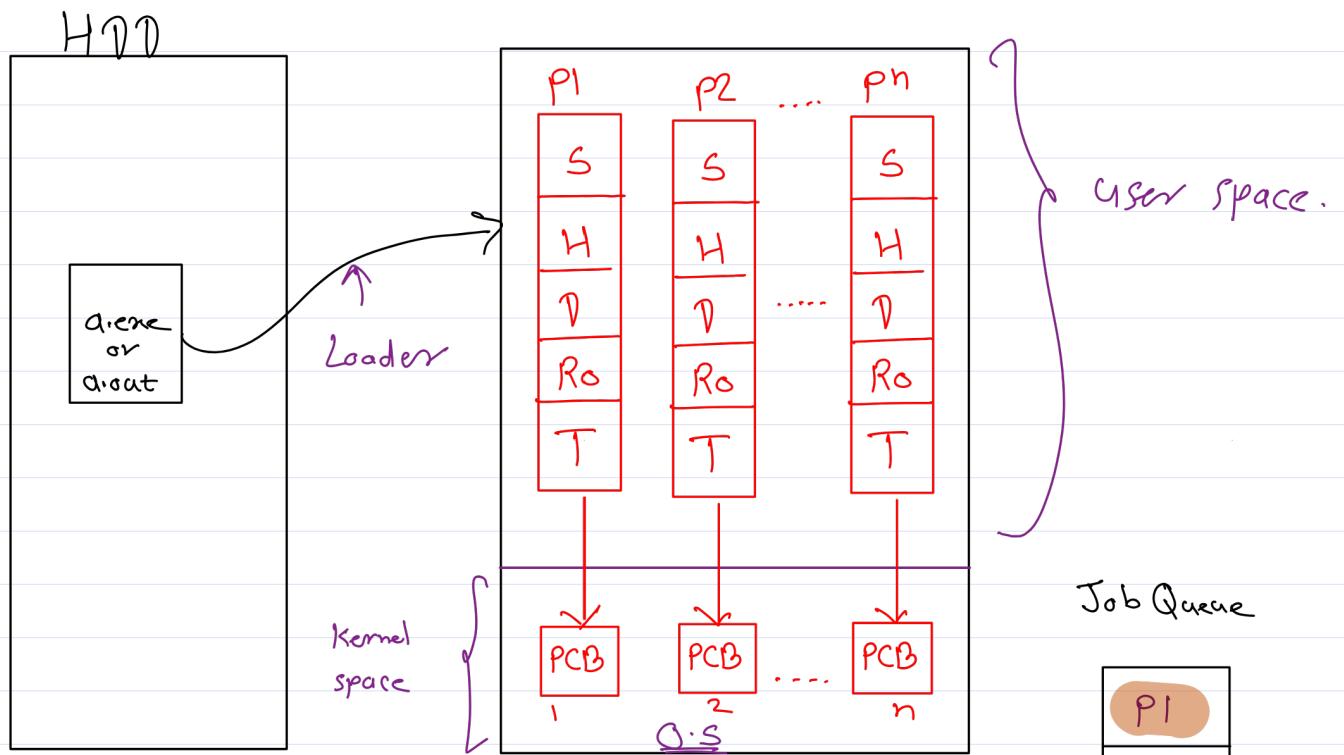
OS



Asymmetric M.P.

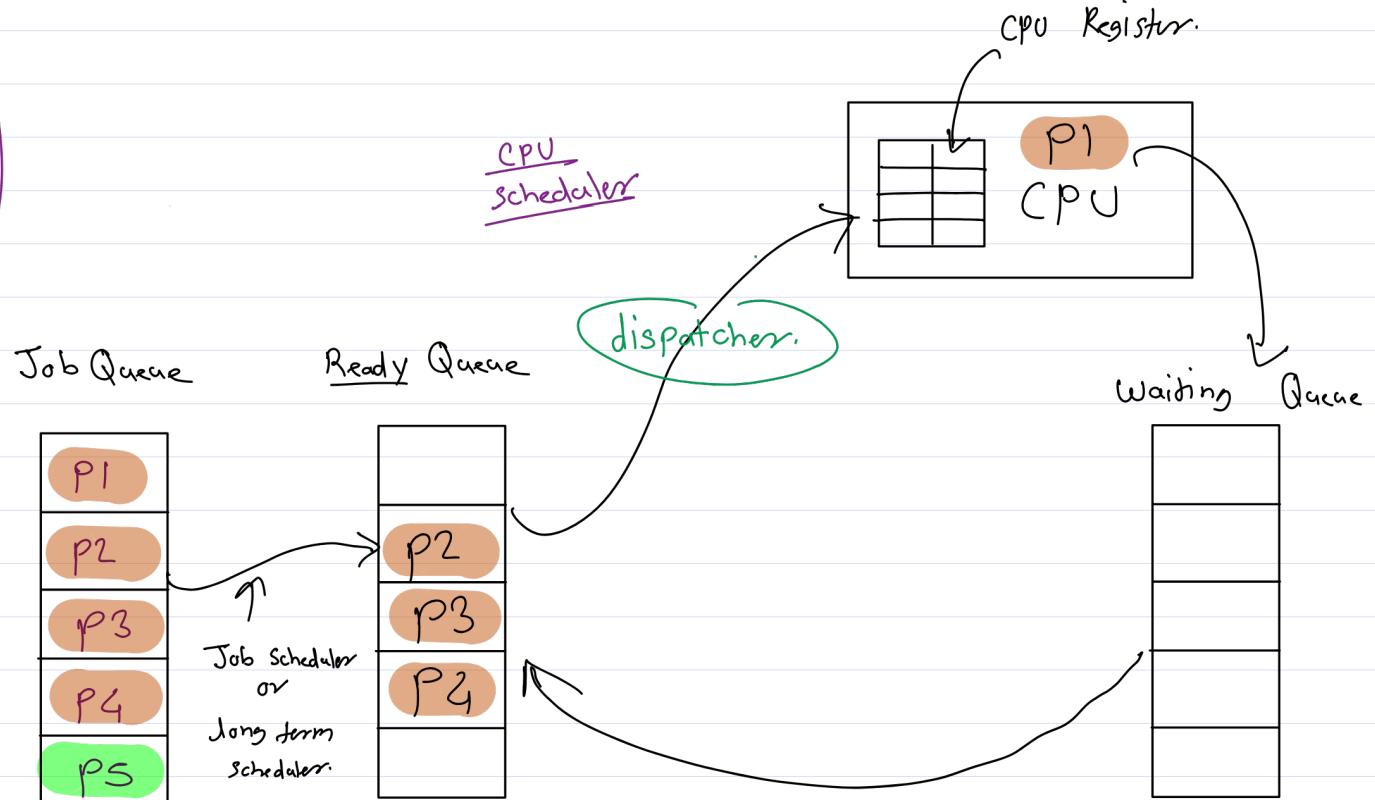
Symmetric M.P.

## \* Process life Cycle



## User Space.

CPU Scheduler : decide which process to Run  
Dispatcher : It Dispatch the selected process.

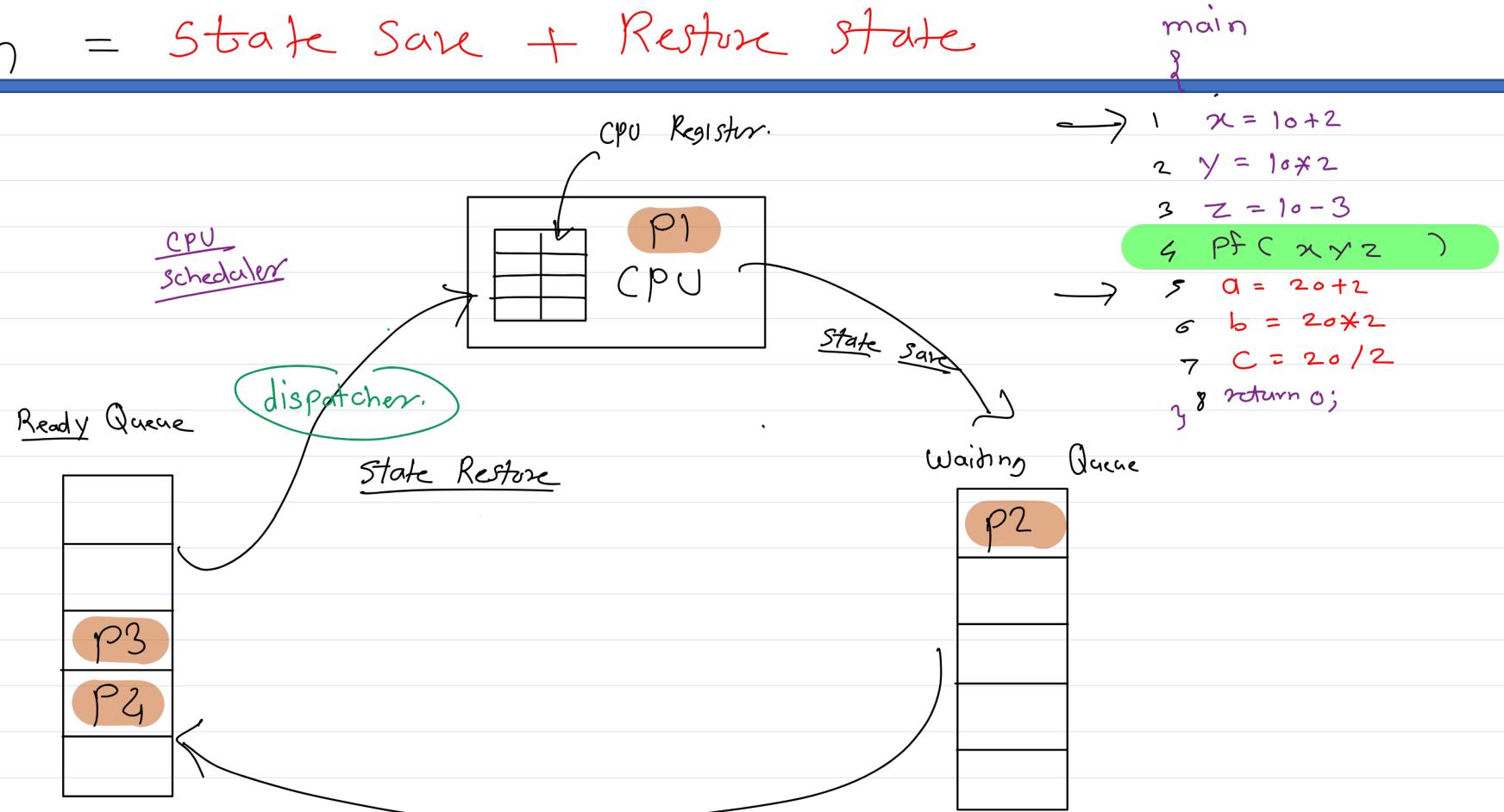


## Kernel data structure.

- ① Job Queue
  - ② Ready Queue
  - ③ Waiting Queue

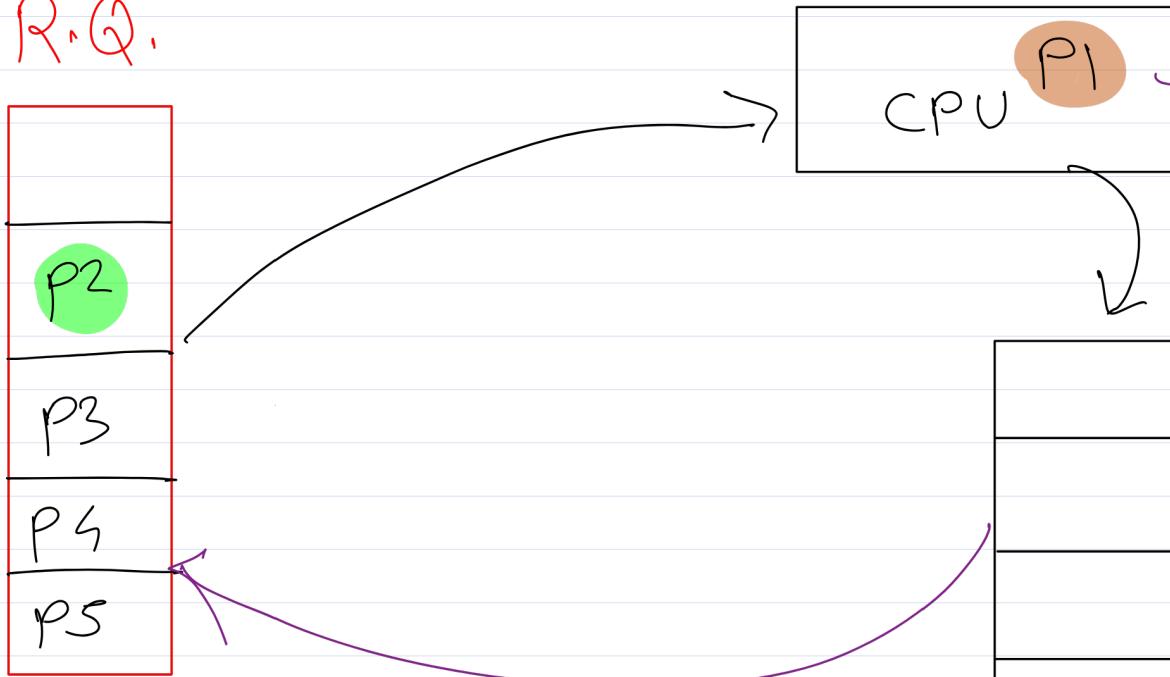
\* Content Switch = State Save + Restore State

- PCB
- Pid
  - Scheduling info
  - Mem info
  - PC
  - Execution Content.



# A) CPU Schedulers

R.Q.



W.Q.

- GT
- PTO
- ① Running → Terminated
  - ② Running → waiting state.
  - ③ Running → Ready state
  - ④ Waiting → Ready.

\* FCFS.

**Convoy effect:** If bigger process arrive first, Avg W.T. increases.

## ►FCFS Scheduling

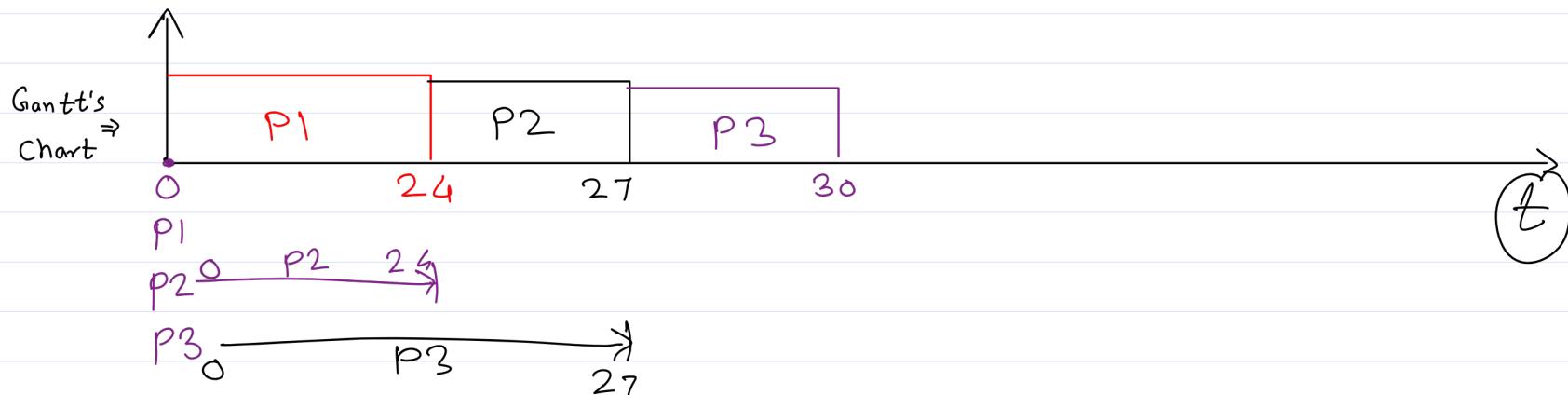


Process	Arrival Time	CPU Burst	Wait Time	Turn Around Time
P1 X	0	24	0	24
(P2) X	0	3	-24	27
P3 X	0	3	-27	30

Avg W.T. =  $\frac{\text{Sum of all process W.T.}}{\text{Num of process}}$

$$= \frac{0 + 24 + 27}{3}$$

$$= \frac{51}{3} \Rightarrow 17$$



Avg. T.A.T. =  $\frac{\text{Sum of all process T.A.T}}{\text{Num of process}}$

$$\Rightarrow \frac{24 + 27 + 30}{3}$$

$$\Rightarrow \frac{81}{3}$$

$$\Rightarrow 27$$

\* FCFS.

## ►FCFS Scheduling

Process	Arrival Time	CPU Burst	Wait Time	Turn Around Time
P1 X	0	3	0	3
P2 X	0	3	3	6
P3 X	0	24	6	30

No Convoy effect.

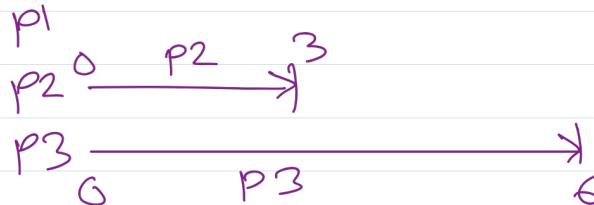
Completion time  
↓

$$\text{Avg. W.T.} \Rightarrow \frac{0+3+6}{3}$$

$$= \frac{9}{3} \\ = 3$$

$$\text{Avg. T.A.T} = \frac{3+6+30}{3}$$

$$= \frac{39}{3} \\ \Rightarrow 13$$



SJF → SJNF (Non-Premptive) ✗  
SJNF (preemptive)?

SJF → gives min Avg wait time.

## ► SJF/SNTF Scheduling (Non-Premptive)

Process	Arrival Time	CPU Burst	Wait Time	Turn Around Time
P1	0	7	0	7
P2	2	4	6	10
P3	4	1	3	4
P4	5	4	7	11



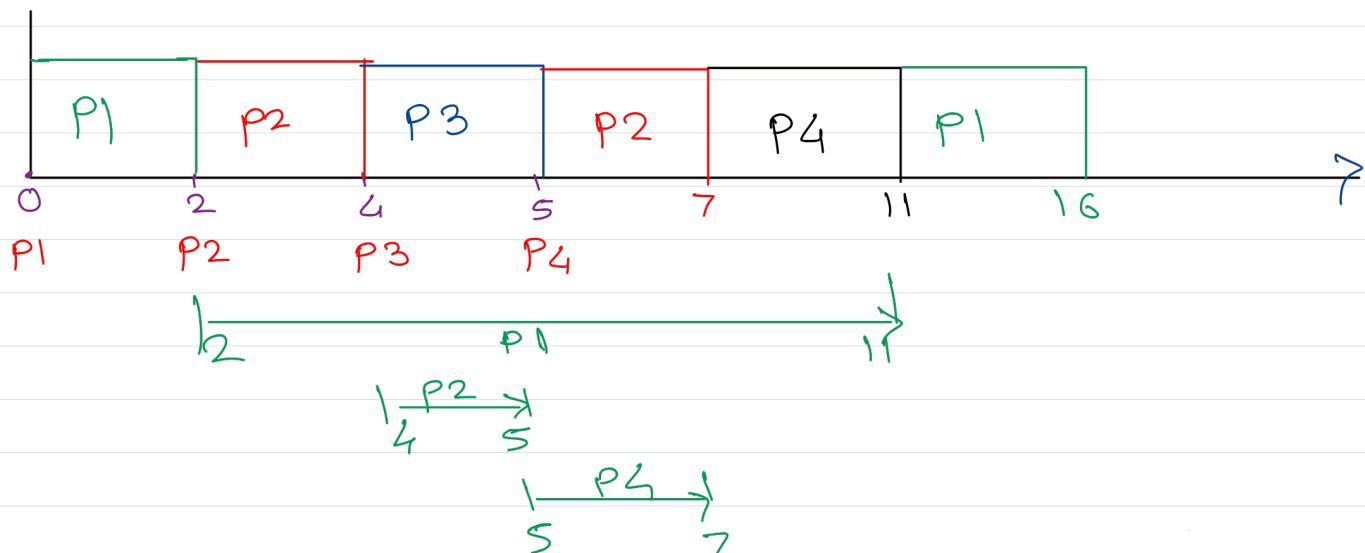
SJF

## ➤ SRTF Scheduling (Premptive)

Process	Arrival Time	CPU Burst	Remaining Time	Wait Time	Turn Around Time
P1	0	7	5 X	9	16
P2	2	4	2 ✓	1	5
P3	4	1	X	0	1
P4	5	4	X	2	6

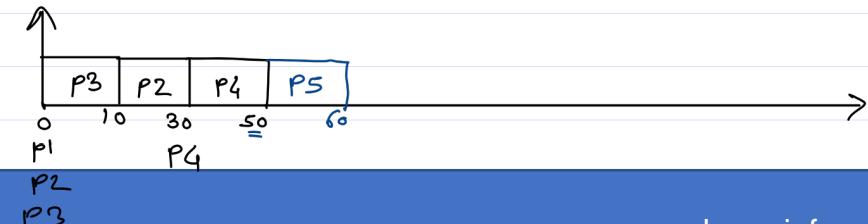
Avg. W.T. =>

Avg. T.A.T. =



\* Starvation      blocked  
 (Starvation)

A.T.	Process	C.B.T.
0	P1	100
0	P2	20 X
0	P3	10 X
30	P4	20 X
50	P5	10



## ➤ Priority Scheduling

min → high priority

max → low priority

Process	Arrival Time	CPU Burst	Priority	Wait Time
P1	0	10 X	3	6
P2	0	1 X	1 - high	0
P3	0	2 ↴	4 - low	16
P4	0	5 X	2	1

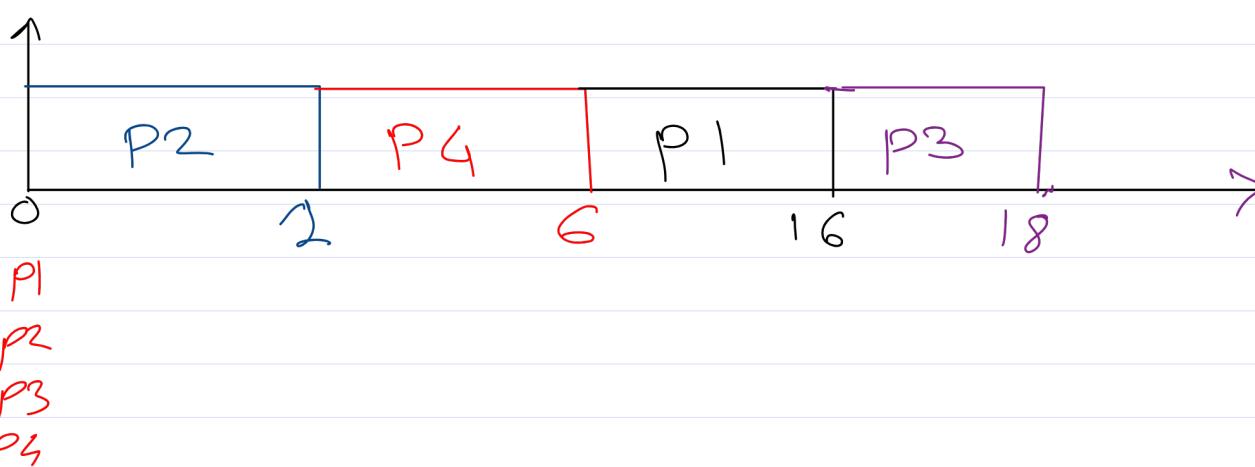
T.A.T

16

1

18

6



\* Starvation  $\xrightarrow{\text{soln}} \text{Ageing}$

blocked (starvation)

A.T.	Process	C.B.T	Priority
0	P1	10	20 $\rightarrow$ 19 $\rightarrow$ 18.
0	P2	10	5 X
0	P3	10	1 X
20	P4	10	7 X
30	P5	10	10



# SJF → Non-Premptive (SJTF)

Process	A.T.	C.B.T
P1	1	7 X
P2	2	5 X
P3	3	1 X
P4	4	2 X
P5	5	8 X

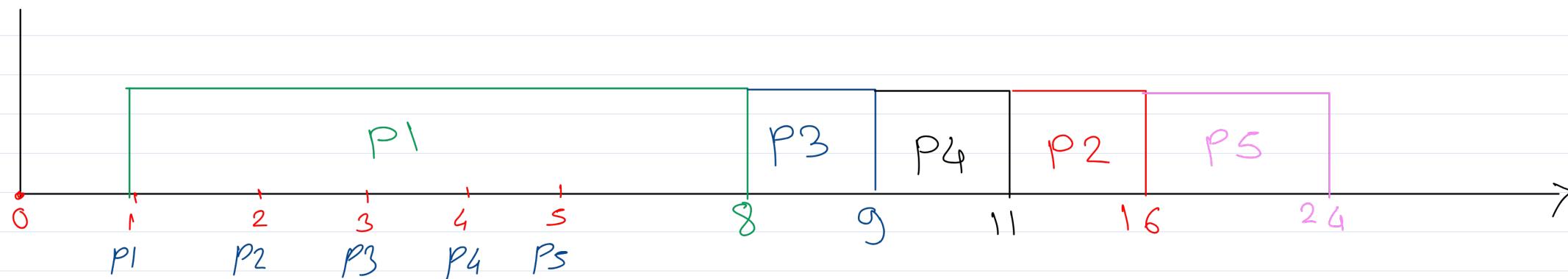
W.T.	W.T + CBT
0	0
5	10
5	15
5	20
11	21

W.T + CBT

TAT

$$\text{Avg. TAT} \Rightarrow \frac{7+14+6+7+19}{5}$$

$$\Rightarrow \frac{53}{5} \Rightarrow \underline{\underline{10.6}}$$



## \* Kernel data structure.

- Job Queue
- Ready Queue
- Waiting Queue.

## \* Process life cycle / process state

- ① New
- ② Ready
- ③ Running
- ④ Waiting
- ⑤ terminating

- ① Ran → terminated
- ② Ran → waiting
- ③ Ran → Ready.
- ④ waiting → Ready.

## \* CPU scheduling Algo-

- ### \* CPU Scheduling Criteria
- CPU utilization
  - Throughput
  - Waiting time
  - Response time
  - Turn around time.

### ① FCFS

↳ non-Preemptive  
→ Convoy effect

### ② SJF

→ min<sup>n</sup> Response time (Avg)  
→ Starvation.

### ③ Priority

↳ starvation  $\xrightarrow{\text{Sol'n}}$  Ageing.

### ④ Round Robin



Given data :  $T \cdot Q \Rightarrow 20$

## ► Round Robin Scheduling (preemptive)

Process	A.T	CPU Burst	Remaining Time	Wait Time	Response Time
P1	0	53	33, 13, X	57 + 24 $\Rightarrow$ 81	0
P2	0	17	X, X, X	20	20
P3	0	68	48, 28, 8, X	37 + 40 + 17 $\Rightarrow$ 94	37
P4	0	24	4, X, X	57 + 40 $\Rightarrow$ 97	57

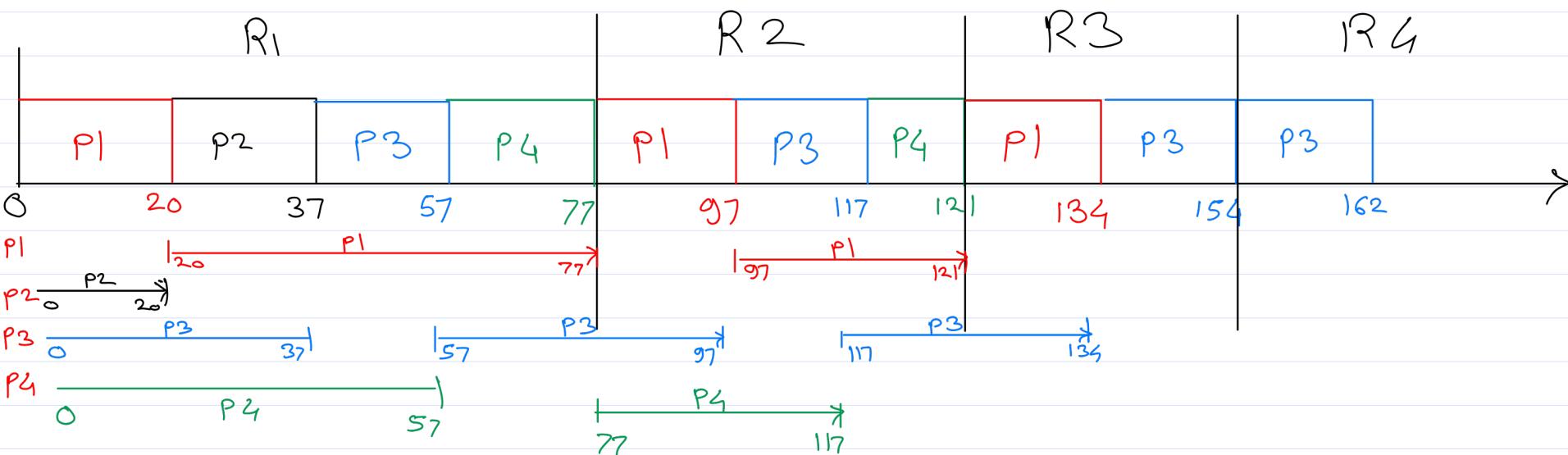
T.A.T.

134

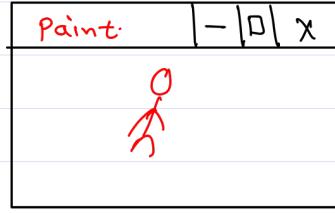
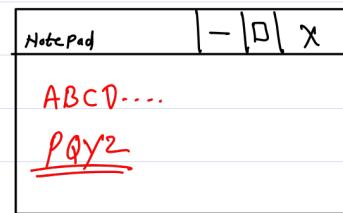
37

112

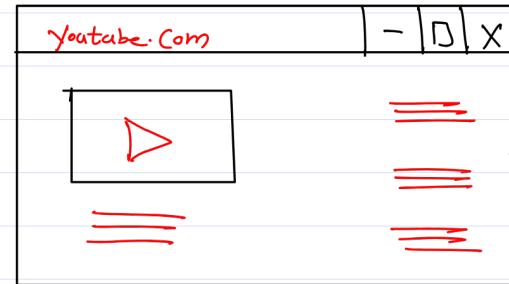
121



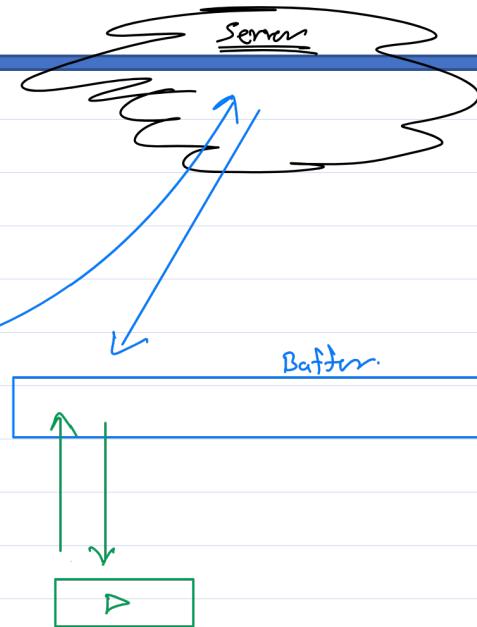
\* IPC



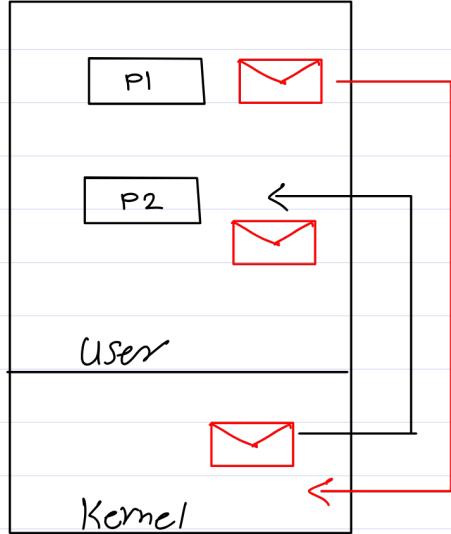
\* Independent Process.



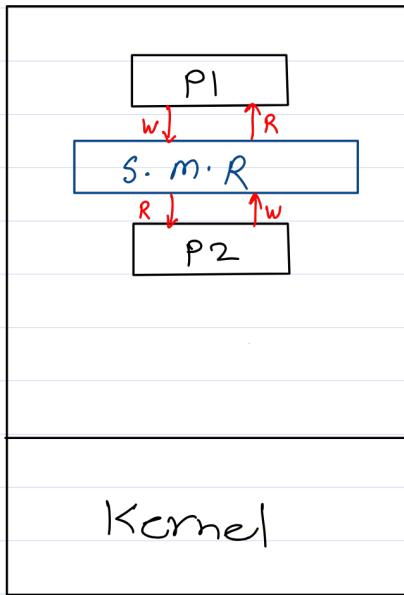
\* Co-Operative Process.



## \* Message Passing

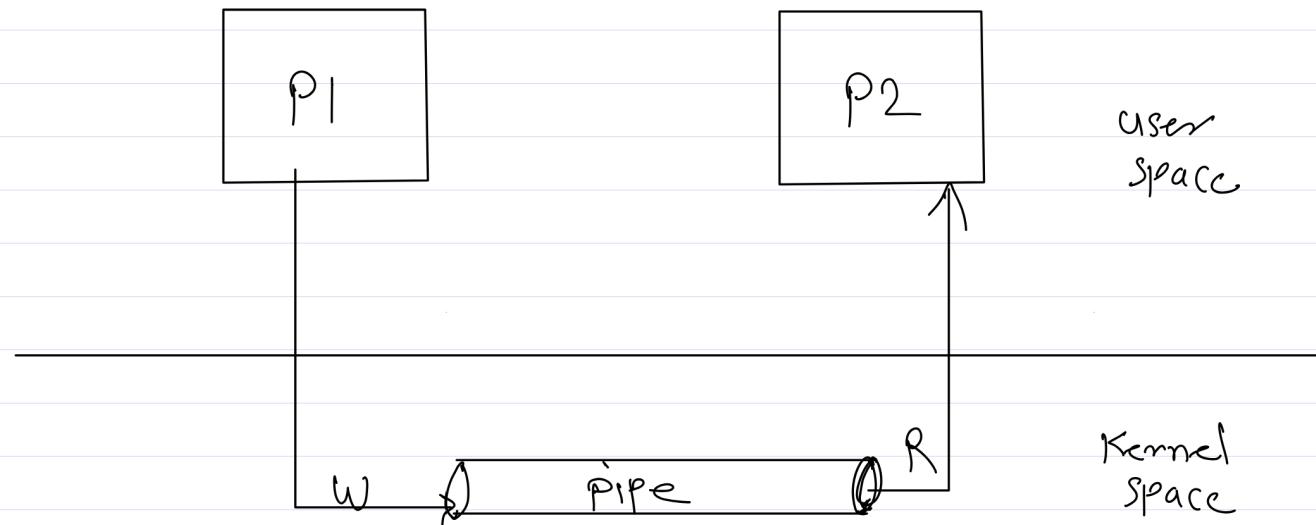


## \* Shared memory.



- ① pipe
- ② message Queue
- ③ signal
- ④ socket

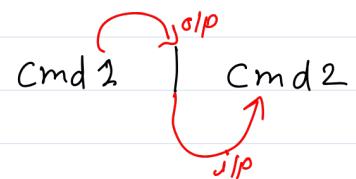
# ① Pipe



\* Stream based  
\* unidirectional comm'.

# ① un-named pipe.

- use pipe Cmd ( )



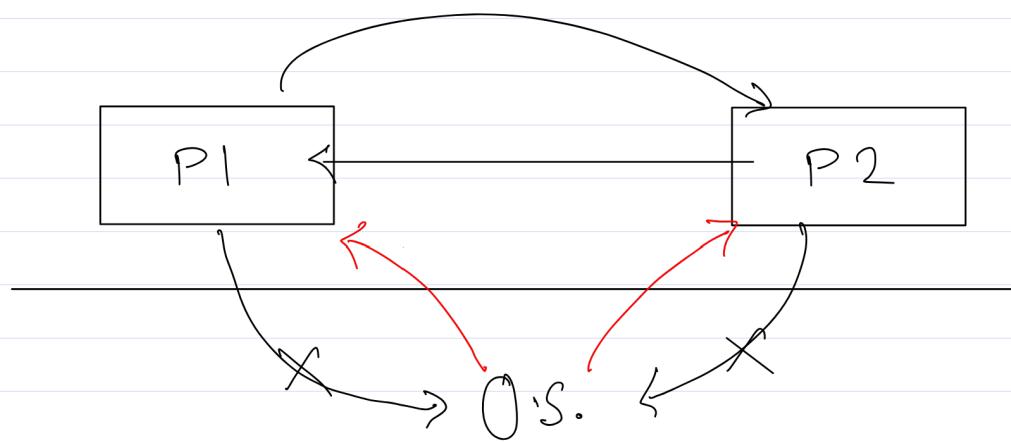
# ② named pipe

pipe( )



②

Signal.

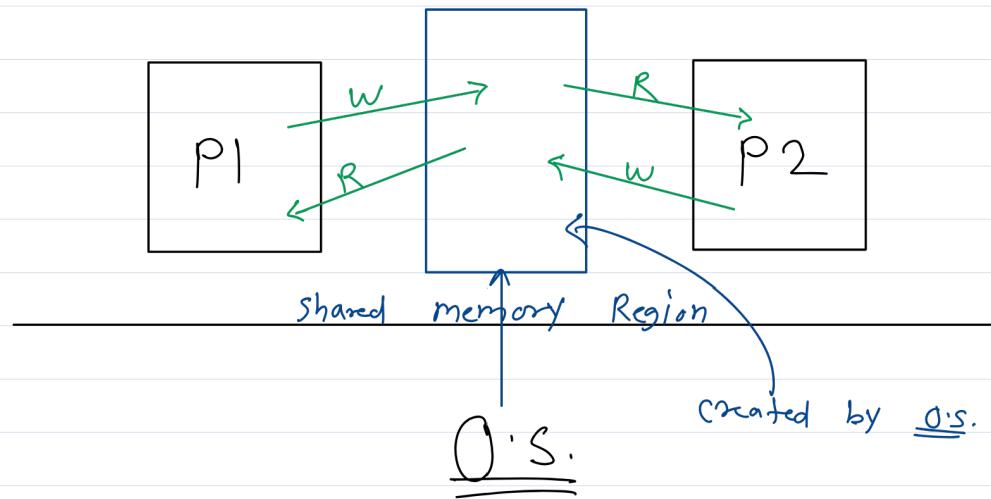


③

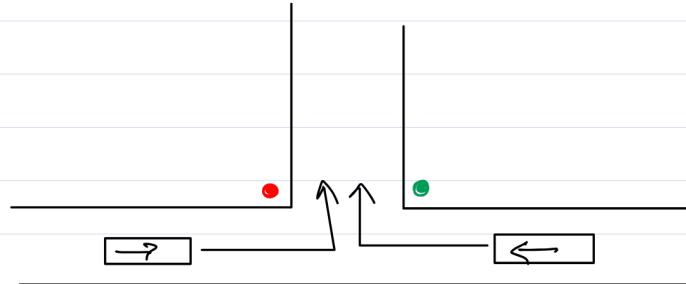
Socket.



## ④ Shared Memory:

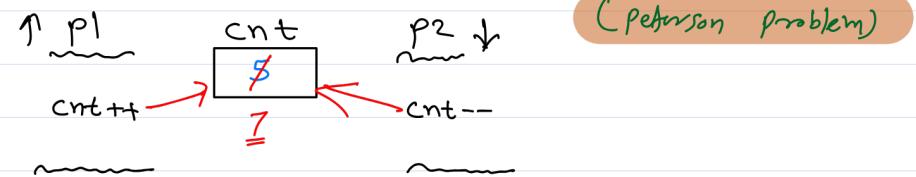


- It is Fastest IPC mechanism.



Race Cond<sup>n</sup>: If two processes try to address same Resource of the same time,

- due to Race Condition data inconsistency happened



Sync mechanism: ensure that only one process will access the Resource at a time and thus data inconsistency avoided.

# ① Semaphore

- It is Counter.

\* Decr op / wait op / P op

- Decrement Semaphore Count

- if  $\text{cnt} < 0$ , True  $\rightarrow$  block the current process.

- Before accessing Resource.

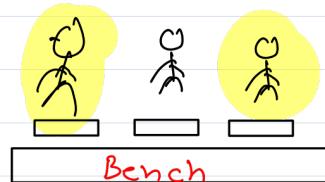
\* Inc op / signal op / V op

- Increment Sema Count

- If one/more process are blocked wake-up  
one of the process.

- After the Releasing Resource.

(↑)  $\checkmark \text{ oprn}$



$P_{opr^n} \downarrow$



cnt  
 $p$

$2 < 0 \rightarrow F$

$1 < 0 \rightarrow F$

$0 < 0 \rightarrow f$

$-1 < 0 \rightarrow T$

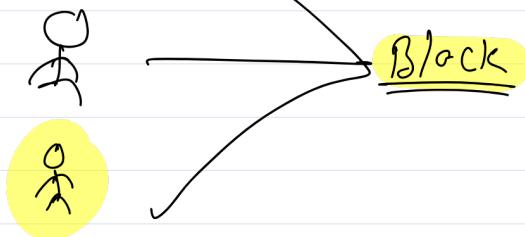
$-2 < 0 \rightarrow T$

$-3 < 0 \rightarrow T$



$\underline{\underline{-2}}$

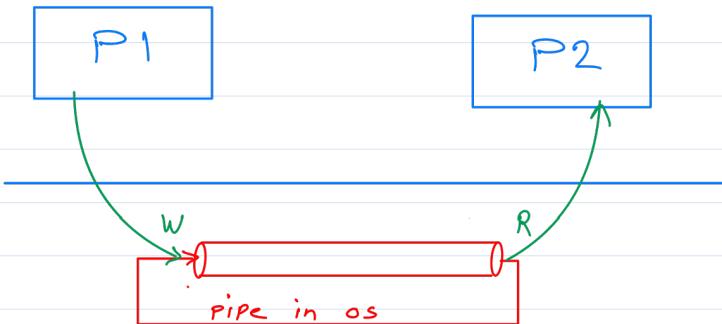
$\underline{\underline{-1}}$



# IPC

## ① Pipe

- stream based
- unidirectional



→ pipe is circular Buffer.

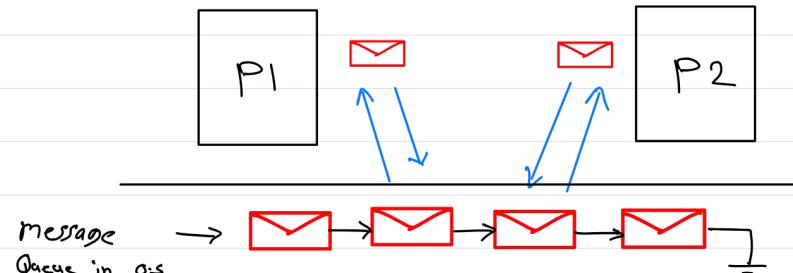
## ③ signal



- Predefined signal
- G4 signal in Unix os.
- unidirectional.

## ② message Queue.

- Bidirectional



## ④ Sockets.



- Socket is comm' endpoint
- Bidirectional
- use to comm' diff or same machine.