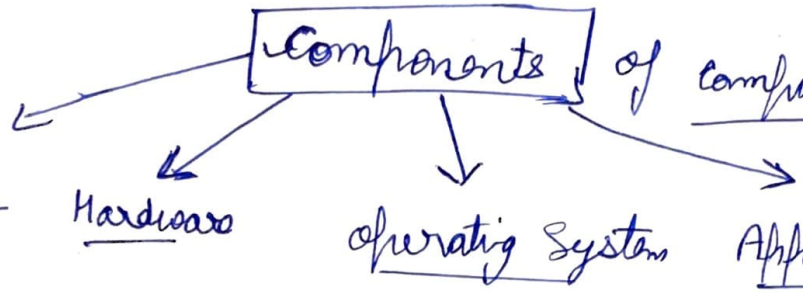
Operating System :-

- OS is a program that acts an intermediary b/w user of a computer and computer hardware.
- OS acts as control program & resource manager.
- " simplifies and manages the complexity of running application programs efficiently.



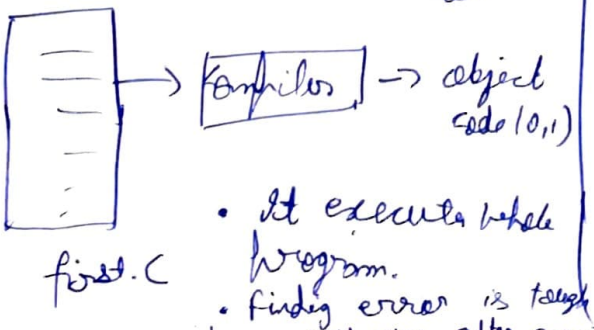
Note :- Diff. b/w compiler and assembler is that a compiler is used to convert high-level programming lang. code into machine lang. on the other hand, an assembler converts assembly level lang. code into machine lang. code.

Lang. Translators

Assembly lang. → assembler → m/c code

Compiler

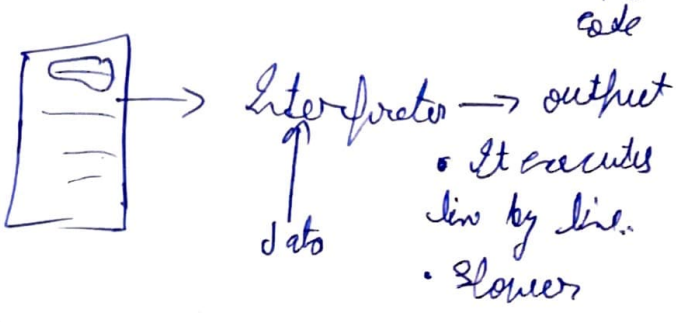
HLL → Compiler → m/c code



- It executes whole program.
- Finding error is tough here as it give after execution.

Interpreter

HLL → Interpreter → m/c code



- It executes line by line.
- Slower

In interpreter, it will convert the HLL code into object code while the program is running means the execution of the program & the conversion or translation done lly. But in compiler, translation done before the execution.

Machine lang:- It is the low level programming lang. It can be represented by 0s & 1s. To overcome this proble assembly lang. is invented.

Assembly lang.:- It is more than low level lang. and less than HLL. It uses numbers, symbols, abbreviations instead of 0s & 1s. for addition, subtraction it uses symbols like Add, Sub, etc.

Types of operating System:-

1) Batch (means batch of job)
2) Multi Programmed *
3) Multitasking *
4) Real time O.S.
5) Distributed
6) Clustered
7) Embedded

• & 11 system

Time-Sharing System.

Batch System:-

Some Important terms :-

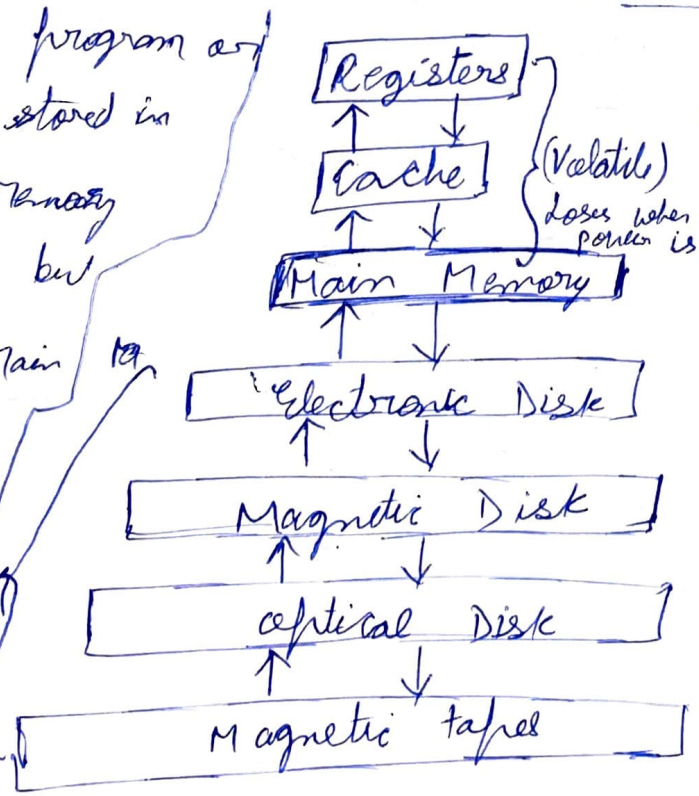
1) **Bootstrap Program** – The initial program that runs when a computer is powered up or rebooted.
   - It is stored in ROM.
   - It must locate and load into memory the OS kernel.

2) **Interrupt** – The occurence of an event is usually signalled by an interrupt from Hardware or Software.
   - HW may trigger an interrupt at any time by executing a special operation called System call.
   - HW may trigger an interrupt at any time by sending a signal to the CPU, usually by the way of system bus.

3) **System call** – SW may trigger an interrupt by executing a special operation called system call. Also known as Monitor call.

→ **Basics of operating System (Storage Structure) :-**

- Basically program and application stored in secondary Memory (HD, SSD) but loads in Main Memory. (RAM)



Non Volatile
Retain its content even when power is removed.

→ Operating System Structure :-

- OS vary greatly in their makeup internally.
- Commonalities :-

i) Multiprogramming – It means the capability of running multiple programs by the CPU.

- Basically multiprogrammed systems provide an environment in which the various system resources (eg - CPU, memory, and peripheral devices), ~~[illegible]~~ are utilized effectively, but they do not provide for user interaction with the computer system.
- MP ↑ the CPU utilisation by organising jobs. So, that CPU always has one to execute.

ii) Time Sharing (Multitasking) :-

- CPU executes multiple jobs by switching among them.
- Switches occurs so frequently that the users can interact with each program while it is running.
- A time-sharing ope. system allows many users to share the computer simultaneously.

Note- A program loaded into memory and executing is called a "PROCESS".

Note:- Multiprog. makes sures that the CPU always has something to execute, thus ↑ the CPU utilisation. On the other hand, T.S is the sharing of comp. resources among several users simultaneously.

→ Batch System :- (1960's)

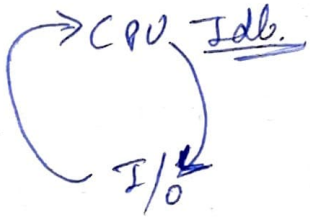
Means similar kind of jobs ka ek batch bnaa lai, aur vo hum computer ko dete hai taki vo execute kar sake.

• In this when a similar kind of job is executing by the CPU, during the execution of 1 program if it requires I/O till that CPU remains idle & instead of switching to next job. In ~~these~~ this there is lot of Idle time to CPU.

Punch cards → operators
Paper tape
Mg. tape

Batches

B_1 B_2 B_3
2 3 4



• After some refinements IBM launched its system in that Monitors are used instead of operator → Fortran

IBSY 7094

→ Parallel System VS Distributed System

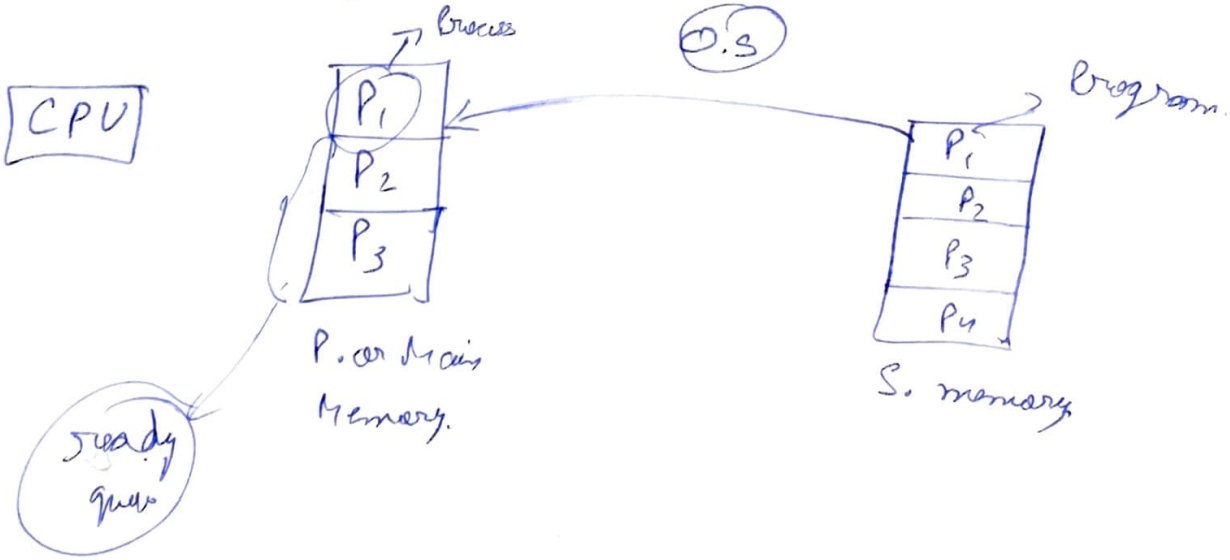
Memory:	Tightly coupled sys. Shared memory.	loosely coupled sys. (means each on having individual kernal, processor) distributed memory. (or) No shared memory.
Control:	global clock control.	No global clock control.
Processor interconnection:	order of Tbps	order of Gbps
Main focus:	Performance, scientific computing	Performance (cost, scalability) Reliability Resource sharing.

System Call :- (Linux has about 300 Sys. Calls)

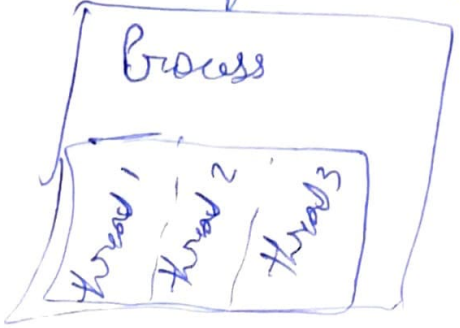
If a user uses any application, API or any program then we use it generally in user mode. But if I want to access any facility of OS then we need to access first kernal Mode. So, for that to go into kernal mode we need System call. It is a programmatic way to shift kernal mode from UM. OS governs all HW devices so, to access those devices we need to access 1st K S.

File Related ⇒ open(), Read(), Write(), close(), etc.
Information ⇒ get Pid, attributes, get System time and data.

→ Process Management :-



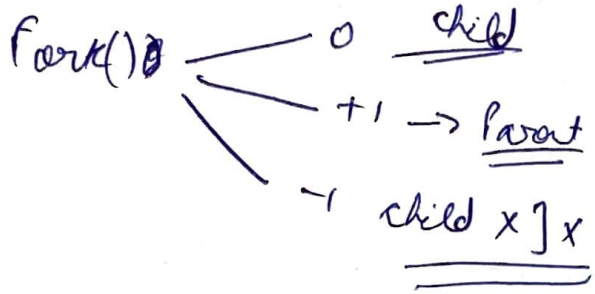
- The program under execution is called as Process.
- A thread is the unit of execution within a Process.

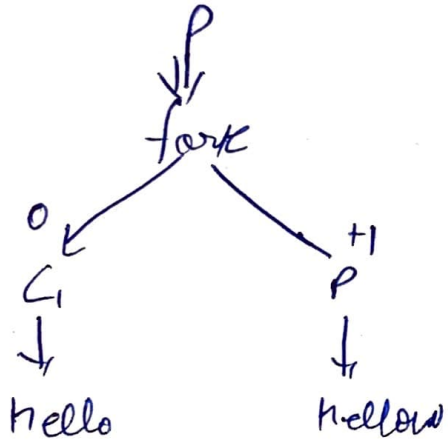


The fork() and exec() System calls :-

Fork(): The fork() system is used to create a separate, duplicate process.

- It is used to create a child process.

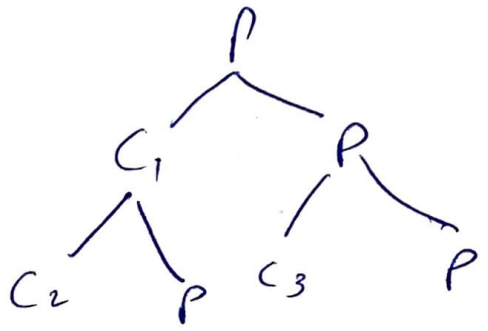





```
main () {
    fork();
    Print f("Hello ");
}
```

```
if fork();
   fork();
```

- It returns 0 as child id



2^n — n → fork.

∴ 2^n, used to calc. total. no of Hello Prnt.

⇒ Total no of child Process generated = $2^n - 1$.

Note:- fork() call se banne walla process jo hain vo, parent process ka exact same hota hai but in exec(), jo process create hota hain (child process) isme other parent mai na ke barabar similarities hoti hai.. ∎

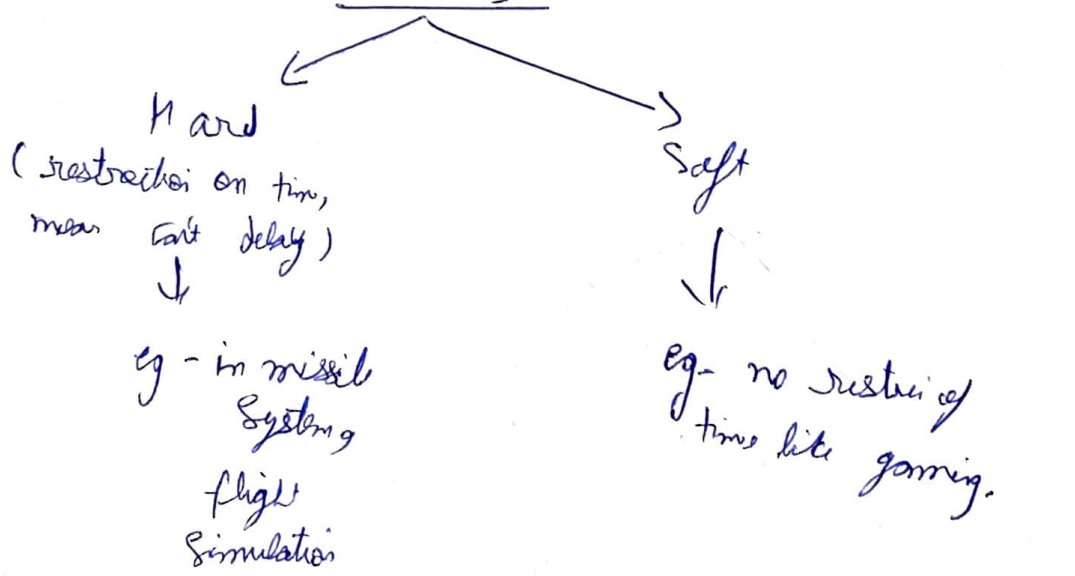
Tab isme basically address space, text segment, data segment, etc. ka inka diff dekhne ko milta hai.

Week - 1 (Requisites).

- Eg. of embedded OS - cell phone, sensors and controllers.
- " Real time OS use - aircraft control, multimedia services

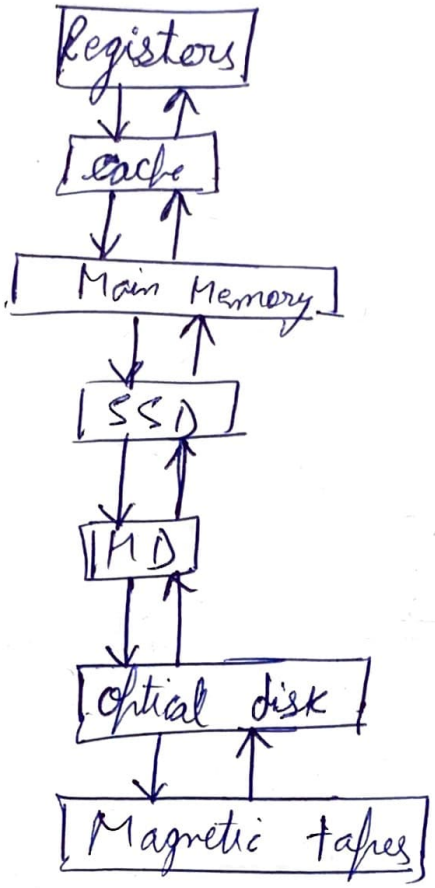
Real - Time OS:-

- eg. like on Utube Streaming to how, what is telling & what is listening is like mean sharing in real time. This is known as real time environment.
- No delays, time matter allot.



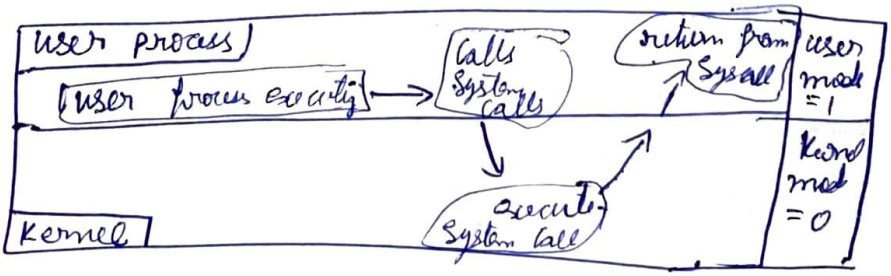
Caching :- Process of copying info into faster storage system.

Storage Device Hierarchy :-



System Call :-

- We use syscall to provide an interface b/w a process and OS.
- A Syscall helps a program request service from the kernel.
- Kernel handles the Syscall.



- Traps are occured by the user program to invoke the func^n of the OS.

OS - Task :-

1) Process Management
   - " creation & deletion.
   - " suspension & resumption.
   - " Synchronization & interprocess communication.
   - " interaction.

2) Memory Management :
   - Allocates & deallocates memory to processes.
   - Manages multiple processes within memory.
   - " the sharing of memory b/w process.
   - Determine which proces to load when memory becomes available.

3) Secondary Storage and I/O Management :-
   - Disk is the primary form of Secondary Storage.
   - OS - perform storage allocation, free-space management.

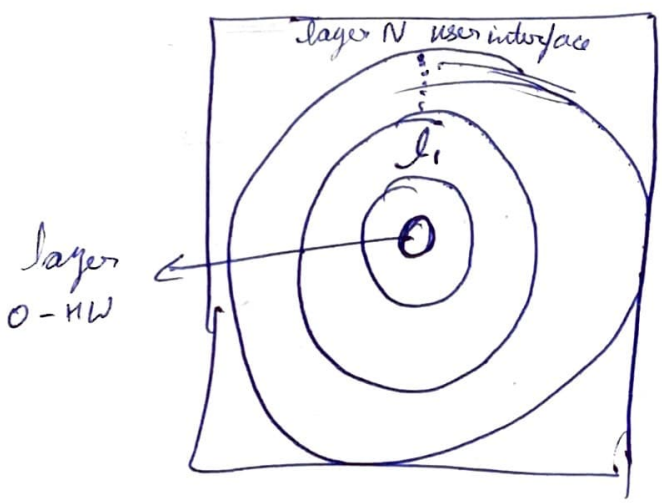
4) File management System :-
   - File creation & deletion.
   - Directory creation and deletion.
   - Supporting primitives for file / directory manipulation.
   - Mapping files to disks (2ary storage).

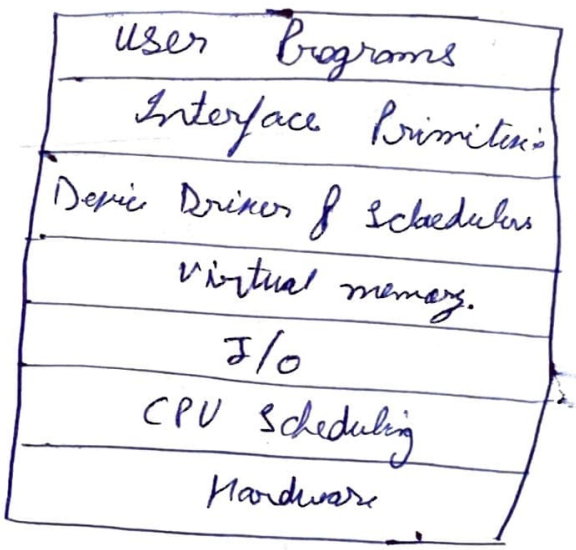
5) Protection And Security :-
   - Protection mechanisms control access of processes to user and system resources.
   - Protection mechanism must :-

1) Distinguish b/w authorised and unauthorised use.
2) Specify access controls to be imposed on use.
3) Provide mechanism for enforcement of access control.

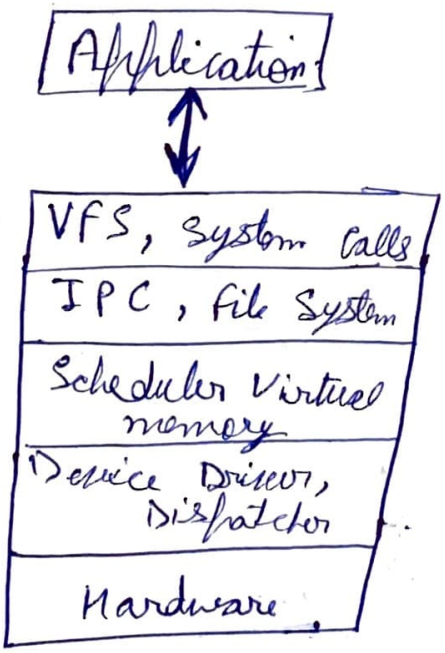
Layered OS Structure :-

- OS divided into no. of layers - bottom layer is HW, highest layer is the user interface.





Monolithic Kernel :-



- More secure

Monolithic larger than Microkernel. Fast { user, Kernel same address space }.

Hard to extend.

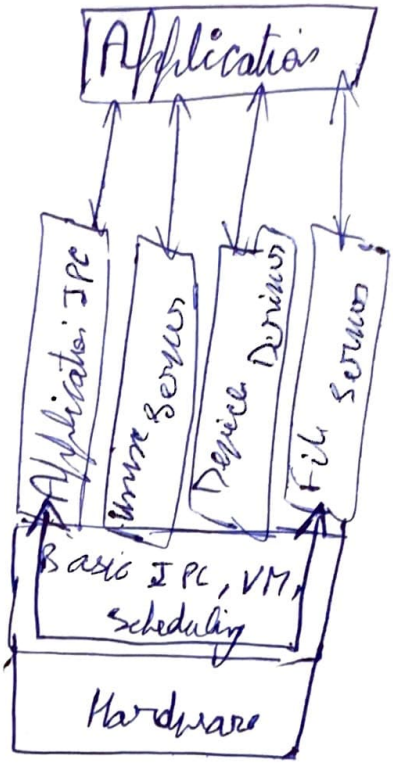
Some exception OS crash.

Linux, BSD, Windows (95, 98, Me), DOS, etc.

Provides CPU scheduling, memory Management, File management through syscalls. ↓ OS fun^{n}.

Micro Kernel:-

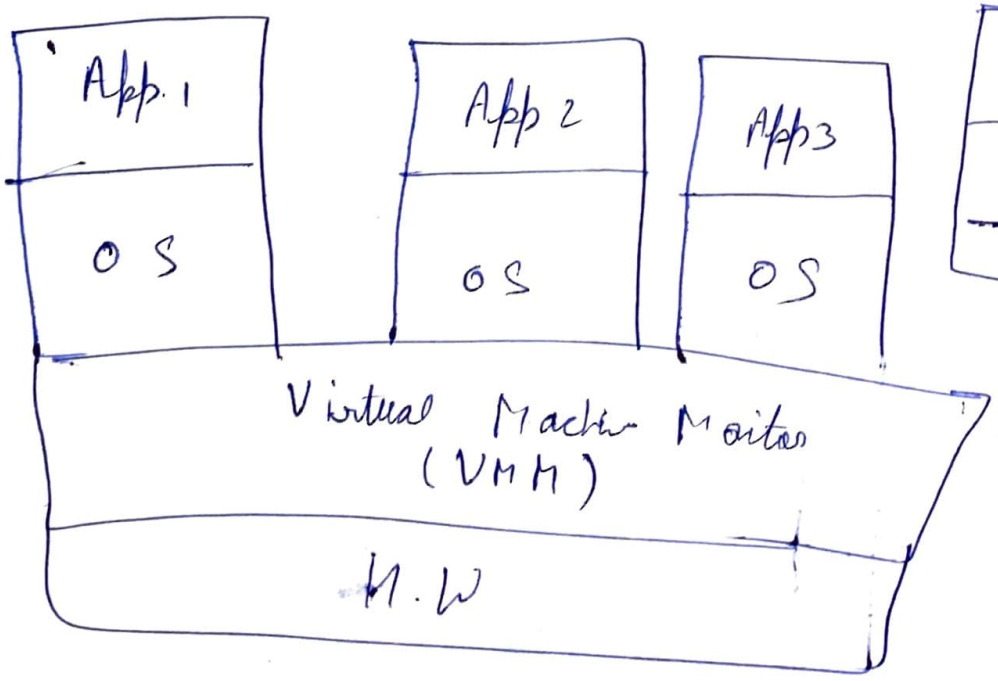
- Separate — Kernel
  - User
- Smaller in Size
- Slow exec^n. (Switching)



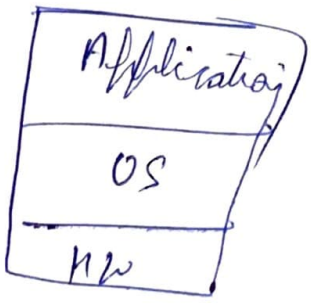
Virtualisation:

- It is the creation of a virtual (rather than actual) version of something, such as an operating system, a server, a storage device, etc.

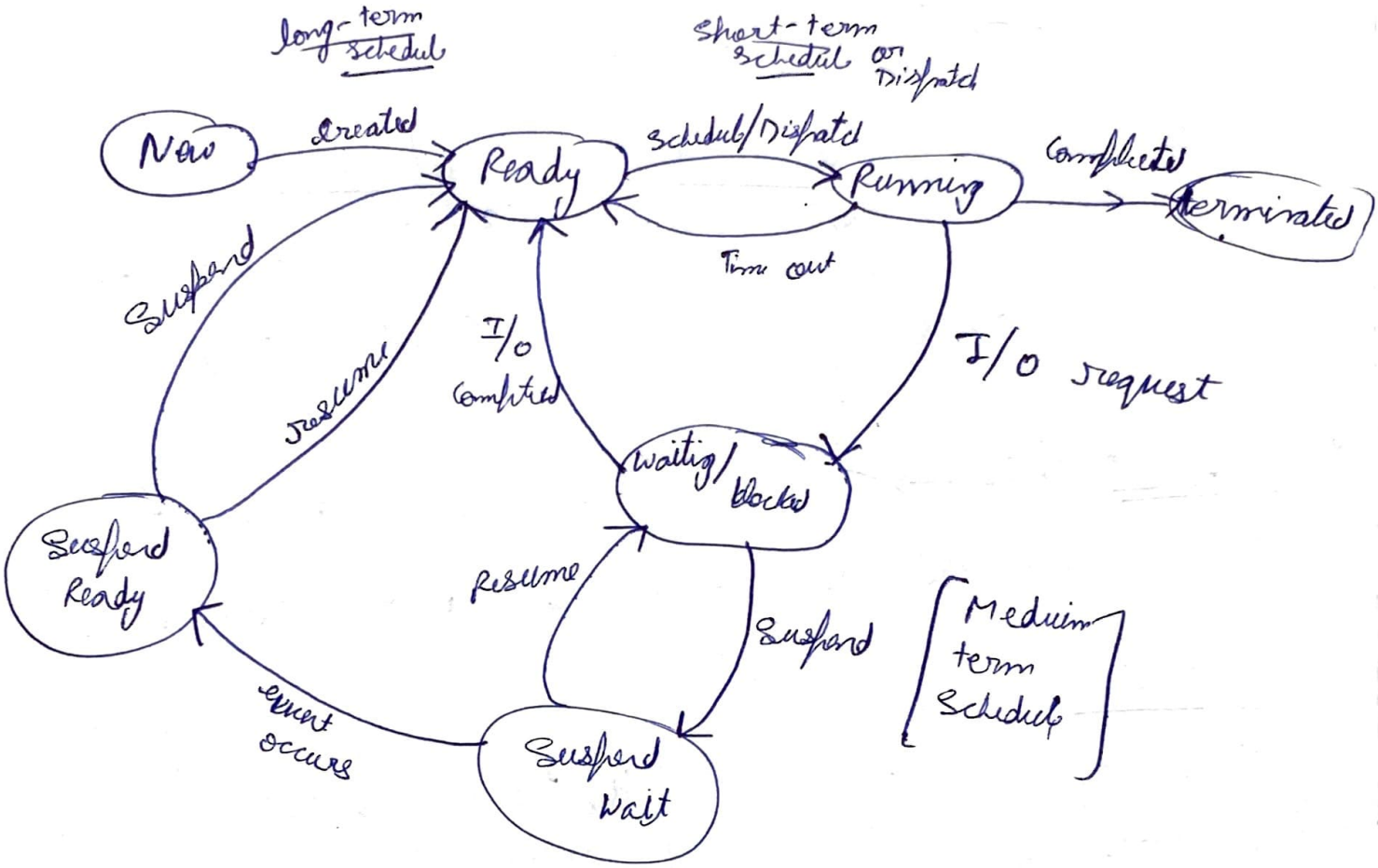
Virtual Machine:



Physical Machine



Process State Transition Diagram :-



Process

- CPU time
- I/O time

ST > MT > LT

- If Process is spending more time on CPU then known as CPU Bound.
- " " " " " " " "
  or I/O time then known as I/O Bound.

Inter Process communication :- (IPC)

→ Series of action

- Process can be Independent or Cooperative.
  - Independent → not affected by execution of others
  - Cooperative → can be affected by other executing prog.

- IPC → → It is a mechanism allows process to communicate with each other & synchronize.

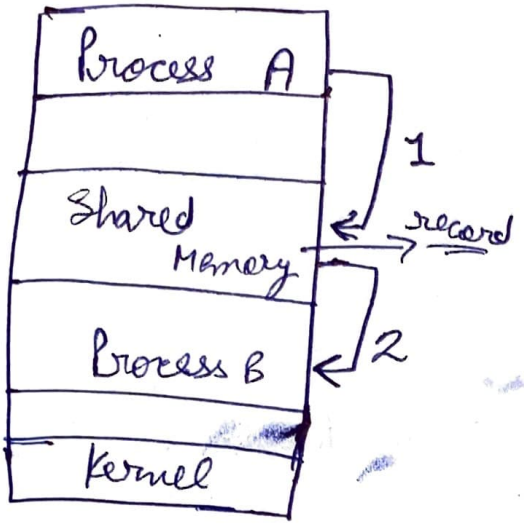
Why IPC :-

1) Info. Sharing
2) Resource
3) Computation speed up. (as Process work together)
4) Synchronisation (help will help to communicate each process to each other).
5) Modularity (Dividing the system func^n into separate processes.
6) Convenience.

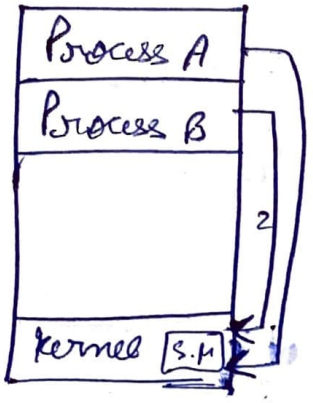
IPC Mechanism :-

1) Shared Memory
2) Message Passing
3) Signals.

① Shared Memory :-



- P_A generates info. & keeps in shared memory.
- So, S.M keeps record of it.
- Now, P_B wants to use S.M so, it 1st checks the record present in the S.M and take the not of info. generated by P_A. and it acts acc. to that. So, it takes another variables or resources.



Problem with it:-

- OS, tries to prevent or process from accessing another's process memory.

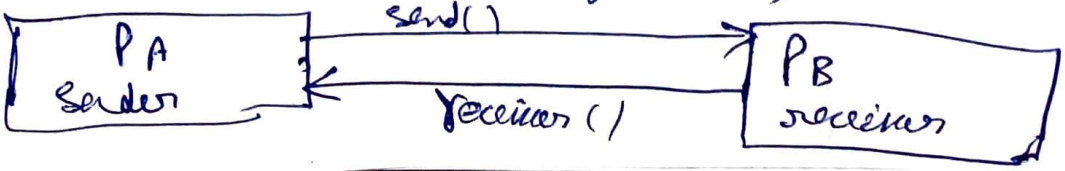
② Message Passing :-

- Here, process communicates with other without using any kind of shared memory.

→ In order to communicate with each other P_1 & P_2 :-

P_1 & P_2 ⇒ ① first establish a communication link.
② Start exchange msg. using basic Primitives.

- Send
  - having - (msg, destination)
- Receive
  - having - (msg, host)



- In this calls, the senders & receivers processes address each other by names.

Mode of communication b/w two process can take place through two modes :-

1) Direct Addressing :-

In this type, the two processes need to name each other to communicate.
This become easy if they have the same parent.
If process A sends a message to process B, then

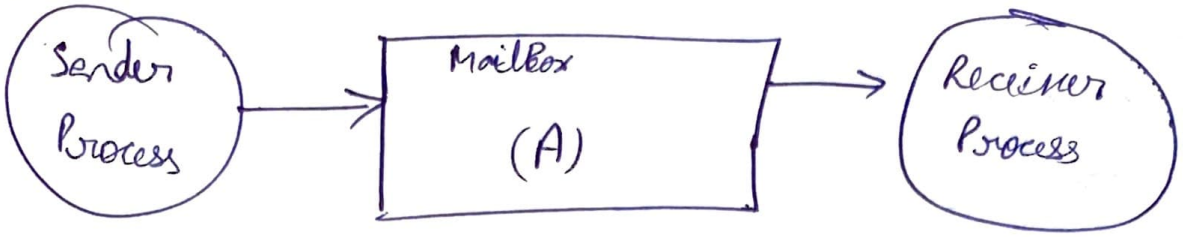
Send (B, msg)
receive (A, msg) } A link is establish

Here, the receiver knows the identity of sender msg destination
This type of arrangement in D. Communication is known as Symmetric addressing
- Usually the link is Bidirectional, but can be made unidir...

2) Indirect Addressing :-

- In this msg send and receive from a mailbox.
A mailbox can be abstracting viewed as an object into which msg's may be placed and from which msg's may be removed by processes.
- The sender and receiver processes should share a mailbox to communicate.
↓
also know as Ports.
- Can be Unidirectional or Bid...

eg:-



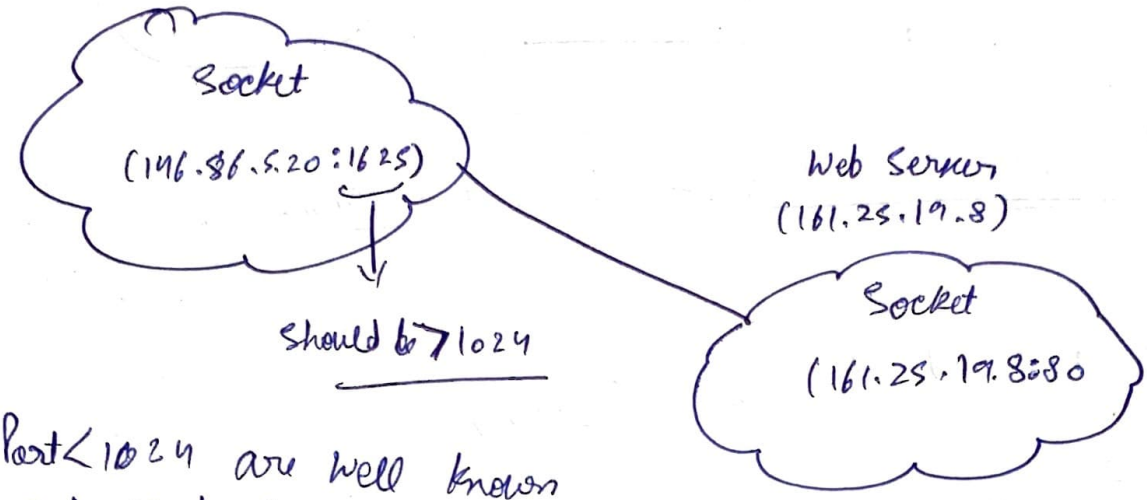
Client - Server System :- Communication.

1) Pipes
2) Remote Method Invocation (Java)
3) Sockets
4) Remote Procedure Calls.

→ Sockets :-

- Used for communication in client - Server Systems.
- It is defined as an endpoint for communication.
- It is identified by an IP address concatenated with a port no.
- Basically, in client - server the client ask info. from server and then server will give that info. to the client. So, Sockets are used to communicate with the server and the server communicate back to the client.
- The server waits for incoming client request by listening to a specified port. Once a request is received, the server accepts a connection from the client socket to complete the connection.
- Server implementing services (such as ftp, http ...).

Host X
(146.86.5.20)



- Port < 1024 are well known and used for standard services

→ Pipe :-

→ also known as ordinary Pipes

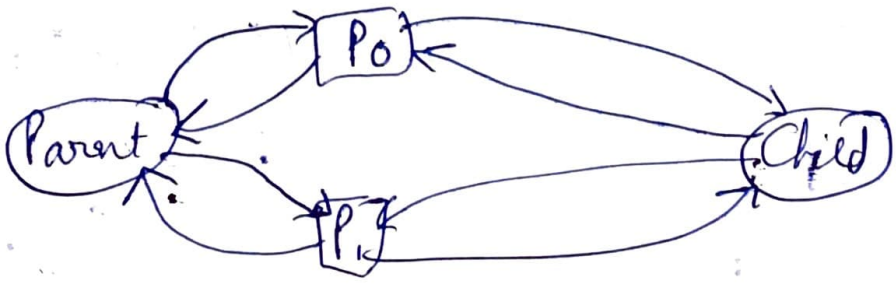
⇒ Child & Parent process communication
⇒ FIFO
⇒ Write 512 Bytes
⇒ Read 1 Byte
⇒ fds = file discriptes

Pipe()

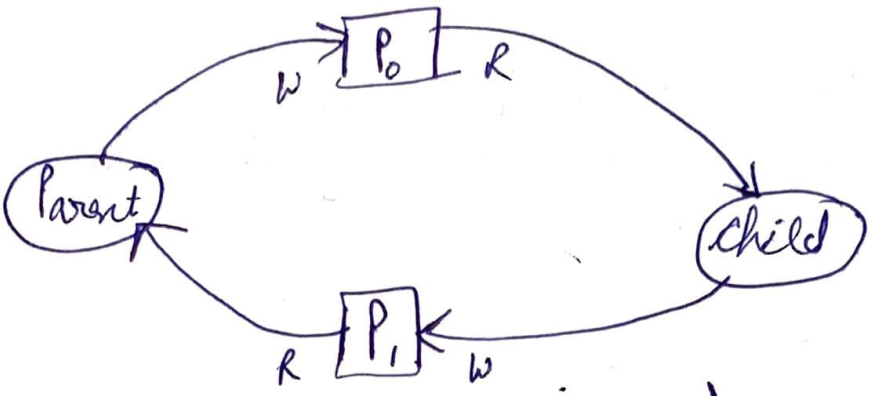
[int pipe (int fds[2])]
fd[0] & fd[1]
Read Write
Return '0' (Success)
" '-1' (failure)
↓
perror():

It is one way Comm.

→ Below is 2 way Comm:



Simplified:-



1) ~~Ordinary Pipes~~ :- Named Pipe

- Bidirectional.
- Referred as FIFOs in UNIX System.

CPU - Scheduling :-

- Scheduling of the CPU is fundamental to O.S design.

→ Arrival time - time at which process enter the Ready queue.

→ Burst time (duration) - time required by a process to get execute on CPU.

→ Completion time - the time at which the process complete its execution.

→ Turn Around time - { completion time - Arrival time }

12 - 11 = 1 hr = 60 min.

→ Waiting time - { turn around time - Burst time }

60 min - 15 min = 45 min

→ Response time - [{ The time at which a process get CPU first time } - { Arrival time }]

→ throughput - no. of processes that are completed per unit time.

→ Single Processor Scheduling Algo :-

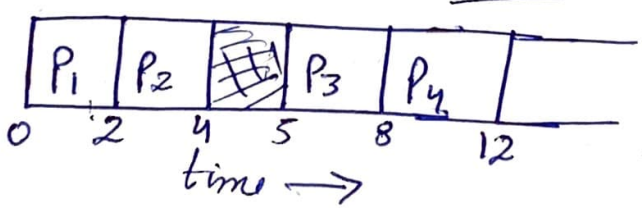
1) First come, First Served (FCFS).] N. Preemptive
2) Shortest Job First (SJF).]
3) Priority] Preemptive.
4) Round Robin (RR)]

① FCFS :-

Process No.	Arrival time	Burst time	Completion time	TAT	WT	RT
P_1	0	2	2	2	0	0 (0-0)
P_2	1	2	4	3	1	1 (2-1)
P_3	5	3	8	3	0	0 (5-5)
P_4	6	4	12	6	2	2 (8-6)
				14		

(Arrival time, Burst time: given)

Gantt chart



TAT = CT − AT
WT = TAT − BT

Note :- in N.P → WT will be same as RT.

∴ Avg. TAT = $\frac{14}{4}$ = .

② Shortest Job First :- (N.P)

Process	A.T	B.T	Start	Wait	F (Finish)	TAT
P_1	0	10	10	10	20	20
P_2	0	29	32	32	61	61
P_3	0	3	0	0	3	3
P_4	0	7	3	3	10	10
P_5	0	12	20	20	32	32
				65		

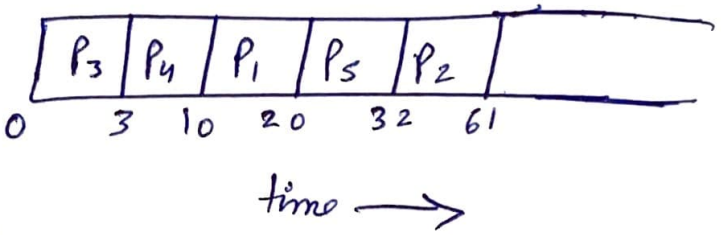
Criteria :- "Burst time" ②

Mode :- "N-P." — ①

TAT = CT - AR

WT = TAT - BT

Gantt chart:



Avg. WT = $\frac{65}{5} = 13$

Avg TAT = $20 + 61 + 3 + 10 + 32 / 5 = 25.2$

→ SJF :- Primptive.

③ Priority Scheduling :-

- Each process has its own priority. out of all available processes, highest priority process gets the CPU.
- If two processor having same priority then use FCFS.

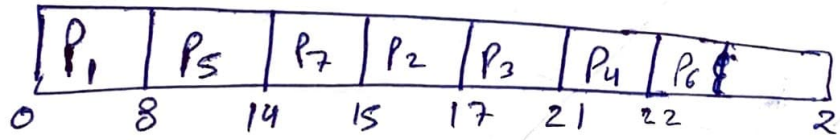
	Priority	A.T	B.T	C.T	TAT	WT	R
P_1	3	0	8	8	8	0	→
P_2	4	1	2	17	16	14	
P_3	4	3	4	21	18	14	
P_4	5	4	1	22	18	17	
P_5	2	5	6	14	9	3	
P_6	6	6	5	27	21	16	
P_7	1	10	1	15	5	4	

less the no. higher the priority

TAT = CP − AT

WT = TAT − BT

→ In N.P same as WT



④ Round Robin :- Preemptive in nature

R.T = CPU first time − AT

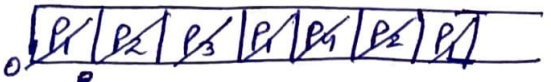
time quantum

Process No.	Arrival time	Burst time	completion time	TAT	WT	RT
P_1	0	~~8~~ 1	~~8~~ 12	12	7	0
P_2	1	~~4~~ 2	11	10	6	1
P_3	2	~~2~~ 0	6	4	2	2
P_4	4	~~1~~ 0	9	5	4	4

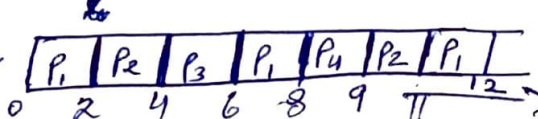
Context Switching

Given :- T.QT = 2

* Ready queue



(Gantt. chart) Running queue



No of CS = 6

T.Q = 20

Q.

Process	B.T	TAT	WT
P_1	53 33 13	134	
P_2	17	37	
P_3	68 48 28 8	162	
P_4	24 4	121	

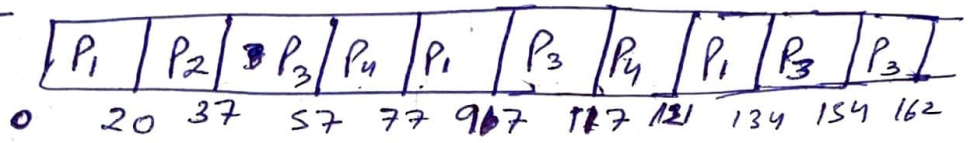
(Y= RR - Easy learning with Nisha)

TAT = CT - AT

WT = TAT - BT

Ready queue - P_1 P_2 P_3 P_4 P_1 P_3 P_4 P_1 P_3

Running " -



Avg. WT ⇒ $\frac{(134-53)+(37-17)+(162-68)+(121-24)}{4}$

$= 81 + 120 + 94 + 107$

Avg. W.T ⇒ $\frac{(0-0)+(77-20)+(121-97)+(20-0)+(37-0)+(97-57)+(134-117)+(57-0)+(117-77)}{4}$

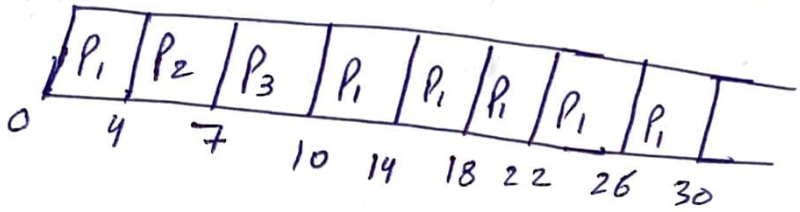
$= 73.$

Avg. TAT = $\frac{134+37+162+121}{4} = 113.5$

Ex-2 ($q = 4$)

Process	B.T	A.T	Start	W.T	finish	TAT
P_1	24 20 16 12 8 4	0	0	6	30	30
P_2	3	0	4	4	7	7
P_3	3	0	7	7	10	10

Ready queue ⇒ 

Running " ⇒ 

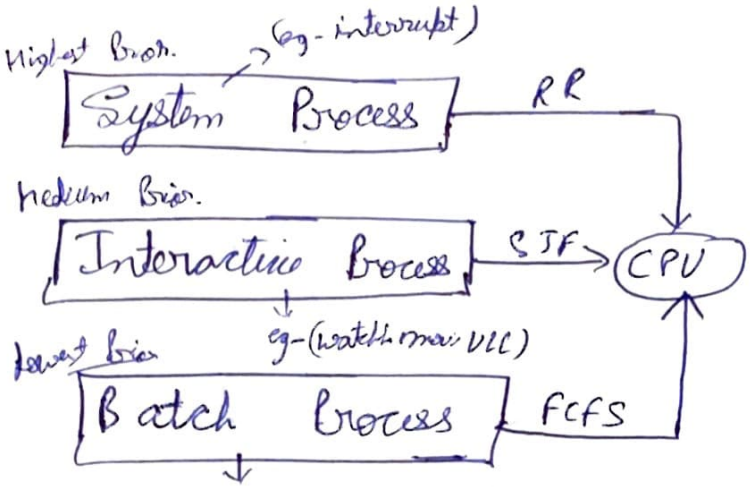
T.A.T ⇒ C.T − A.T

Avg. TAT = $\frac{10 + 7 + 30}{3} = \frac{47}{3} = 15.67$

WT = TAT − BT

Avg. W.T →

Multi Level queue scheduling :-

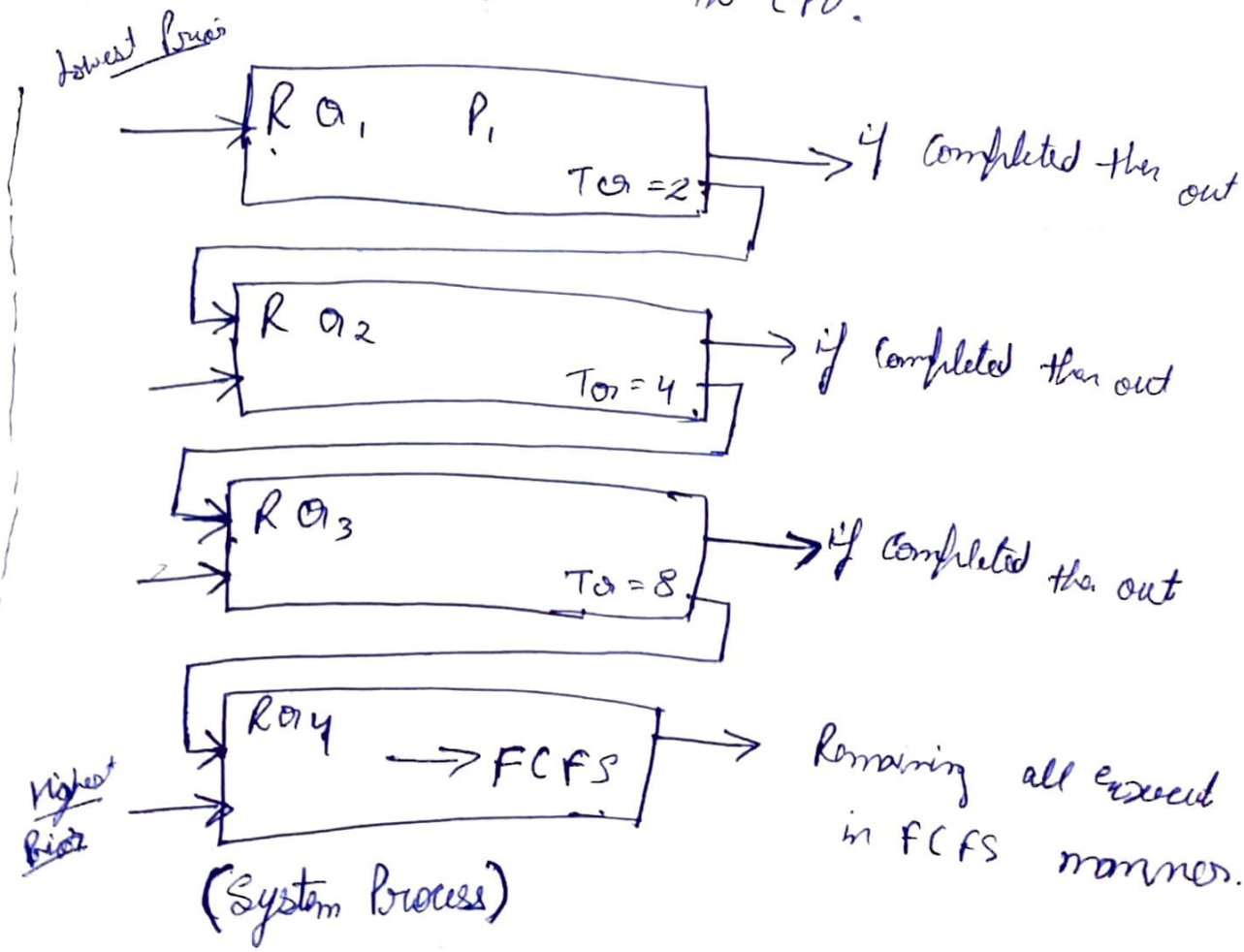


☆
- Here, is problem of Starvation mean if higher prio. Process keep coming than lower or medium prior process will not get execute soon, / often lot of time.

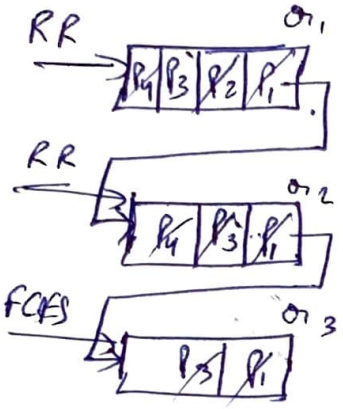
- Interactive process ≅ foreground. (RR algo.)
- batch process ≅ background. (FCFS algo.)

Multi Level feedback queue :

- We use this feedback queue for lower prior. of Process so, that they also get to the CPU.



Q- MLFQS:-



Process	B.T	A.T	C.T	T.A.T	W.T
P_1	~~53~~ 36 11	0	136	136	83
P_2	~~17~~ 0	0	34	34	17
P_3	~~68~~ ~~51~~ 26	0	162	162	94
P_4	~~24~~ 7	0	125	125	101

Q_1 = RR = 2(17)

Q_2 = RR 2(25)

Q_3 = FCFS

Gchartt chart:-