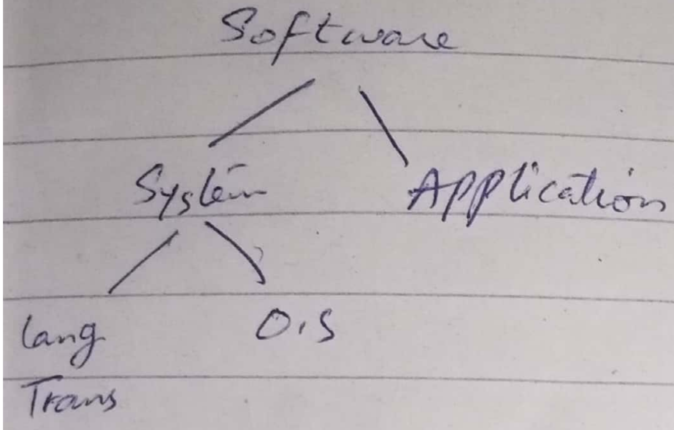
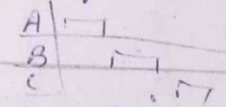
# OPERATING SYSTEM



1. Communication bridge.
(O.S is connect user with hardware)

2. Control program/ ~~bitee~~ through put.
(one time run many program)



*) Running: one time run one program, Resume: again go same prog

3. Resource allocation.

*) Resource: I/O device, memory, O.S C.PU k
bi resource u
icrta h.

$$\text{Utilization time} = \frac{\overset{\text{service Time /}}{\text{Execution Time /Burst tim}}}{\text{Turn around time}}$$

Smallest unit in O.S is "Quantum"
Quantum size "how many quantum"

Scanned with CamScanner

- Execution Time decided to O.S.

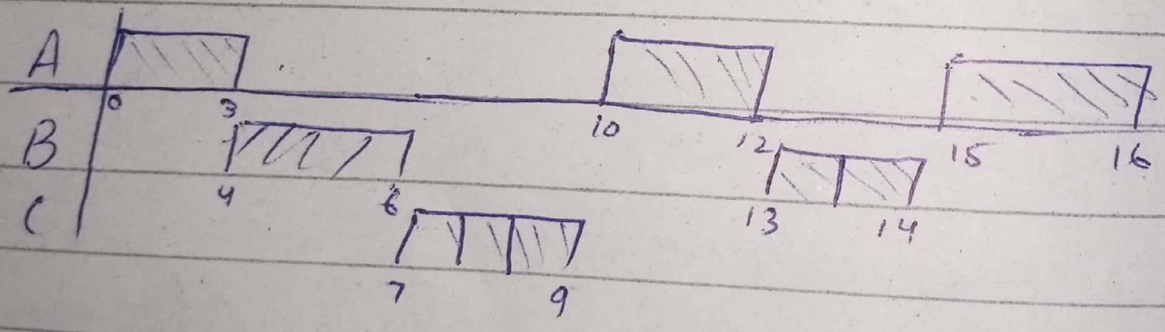
- Time around: Finish time - Arrival tim

T.T

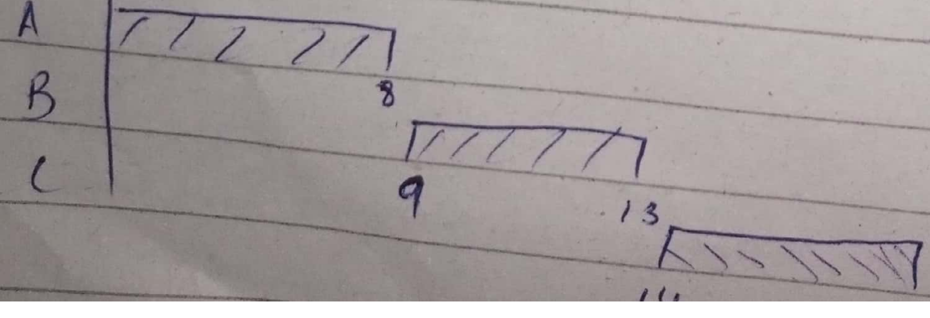
| Averag T.T | | Execution | Arri | End |
|---|---|---|---|---|
| 8/16 = 50% | A | 8 | 0 | 16 |
| 5/13 = 35% | B | 5 | ~~4~~1 | 14 |
| 3/7 = 42.8% | C | 3 | 2~~4~~ | 9 |

q = 3

## Round Robbin:-



## Fist Come Fist Seive:



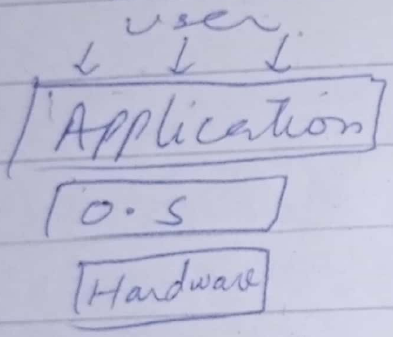
|   | E.T | A.T | End time | T.T | A.V e/T |
|---|---|---|---|---|---|
| A | 8 | 0 | 8 | 8 | $8/8 = 100\%$ |
| B | 5 | 1 | 13 | 12 | $5/12 = 41.6\%$ |
| C | 3 | 2 | 16 | 14 | $3/14 = 21.4\%$ |

$$U = (100 + 41.6 + 21.4) \div 3.$$

$$\boxed{U = 54.3}$$

O.S:-

Interact b/w hardware & user



Q Why do we need O.S?

A If we want hardware without O.S, we need to write program again & again.

- Through put (many task per unit time)
e.g. Linux

- Convinence: easily acquire hardware
e.g. Window

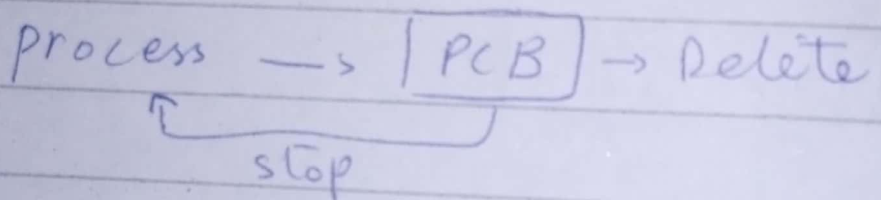
Scanned with CamScanner

# Process Control Blocks:-

O.S create PCB in every process.

It is data structure and store all process data.

It has (Process ID) of each proces



When process stop/end, PCB life time are delete.

1. Process ID
2. State (Running, Ready, Wait, New, .....
3. Pointer (To point the parent process)
4. Priority
5. Program Counter{which next inst execute)
6. C.P.U register (during execution info in it
7. I/O Information
8. Accounting Info (time limit, execution time

Scanned with CamScanner

## Batch Processing:- 1960-1970.

Simple B.P (Buffer)
Multi B.P (Spool )
Time sharing

| $J_4$ | $J_3$ | $J_2$ | $J_1$ | —> | C.P.U | <—> | Buffer | <—> | I/O |

Batch
Uni processor,

Buffer is temporary storage
Disadvantage:- Idle state.

## Multi B.P:-

Naturally multi ~~processing~~. programming. In a same batch input gives by $J_1$ and output generate $J_1$ during the processing it takes input $J_2$.
Like Printer:-

## Time Sharing:-

Uni processor.
After 0.2 sec watch second job is coming or not. & compare multi job, but which have large memory in RAM.

S.1S

| Monitor |
|---|
| Job 1 |
| Job 2 |
| Job 3 |
| Job 2 |

RAM
$Job_1$=5
$Job_2$=7
$Job_3$=

If 2 Job have same size. first run which is arrival first.

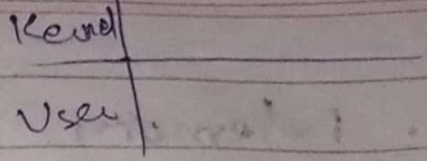
## Multi Processor:-

Flynn's arch. —> e.g. DOS.
SISD (single inst single data). ∴ SI: One pro
SIMD e.g. current O.S: windows, Linux
MISD It is not physically exist. but it is in architecture.
MIMD (Multi inst multi data).
" ~~#~~ ~~os~~ current O.S.

Distribution archi & shared (MIMD)
It works in big data

multicomp archi
↓
It is Distributed

multi proc OR (SMP). Simitric multi processor
↓
It is share



## Kernel:-

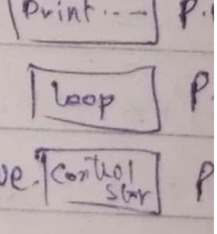
Mono Kerrel arch (Unix, Linux)
Micro Kernel / Layared arch (Windows.

-> Module:-
Main processor } Reside in Kernel.
service " } in Mono kernel.
utility " }

6 layers:-

It is enchace able.
means any program in hardware automa
in Kernel.
It can be communicated.

## 0 Layer (Process Creation):-



break in to the process.
It come to the ready queue.
parent child relation. If parent
close all child will be closed.

P=5 ~~P.5~~ Parent | depends on
P=5.1 child | piority
P=5.2 Child |

Comm
· M
○ Process
User

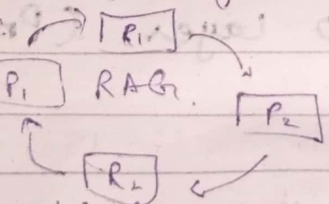
## 1. Memory Managment:-

It is like cache or RAM.

## 2. Process Communication:-

IPC (Inter Process Communication)

- **Dead lock:-** When 2 thing (process) have only one resource both process are fight, take who which have high piority & second one in waiting and ~~RAG (Resource Allocation Graph)~~ both are not executing. after that it take starvish lock.



- Process communication share our PCB to find which process should ~~have~~ resources
- It has three types.

Fully aware to each other
Partial " " " "
Unaware " " " "

- **Fully Aware:-** It share fully PC
It does not have dead lock.
- **Partial Aware:-** It share partial PCB. It has chance to dead lo
- **Unaware Aware:-** It does mot share.

$$\begin{array}{c|cc} & R_1 & R_2 \\ P_1 & 1 & 0 \\ P_2 & 1 & 1 \\ P_3 & 0 & 1 \end{array}$$

Dead lock
Dead

## 3. I/O Managment:-

I/O

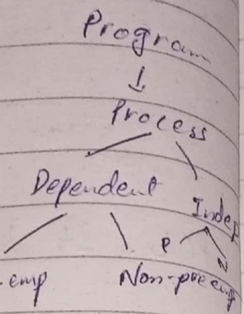
## 4. User Program:-

## 5. Interface:-

# Ch#3 Process & Discription

process: program in execution mode or set of instruction.

It has two types.

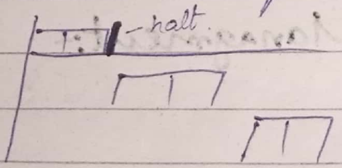


**Dependent:-** program is dependent: e.g. if else condition for loop.

**Independent:-**
e.g. print command.

**Pre-Emptive:-**

If any process halt by interrupt or by force.
It can be on ready queue.



**Non-premptive:-**

Does not halt.

Operating system or Kernel

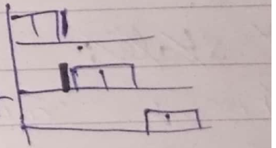
Operating System Do job when It process:

1. Process creation. (reason) login
to open new program
parent create child
o.s reason

2. Process Termination. It has 14~~5~~ reas
1. Compelete process
2. Dead loc(c
3. Starvish
4. When parent finish childs also finish
4. Parent request to finish child number

3. Process Block: It has only 1 rea Software Interupt. or If any process ~~dep~~ demand of resource it is in blocked state then in ready q
or after block it goes in suspend. multiple to depend on event call

4. Process Resume:-



The second process start is called resume.

5. Communication.

6. **Suspend:-** RAM overload it tak hardware some part (virtual memory) behave Like temporary memory. or sway

resource occur = event.

2. It is a time chain. wait for resource

3. Parent suspend to child.

4. O.S reason

## Logical Views Of Process OR Process Stalé Model:-

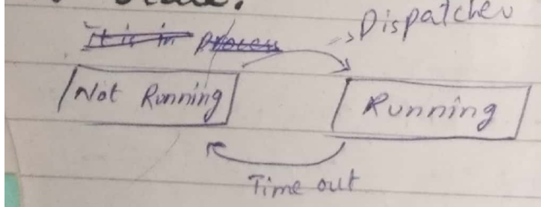
- 2 stalé process model.
- 
- 5 stalé process model
- 6 " " "
- 7 " " "

### 2 Stale:-

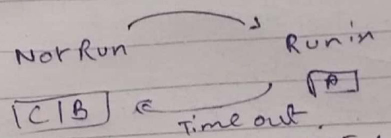
Process switching take more time of O.S.

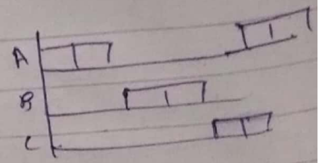
### 2 stalé:-

~~It is in Process~~



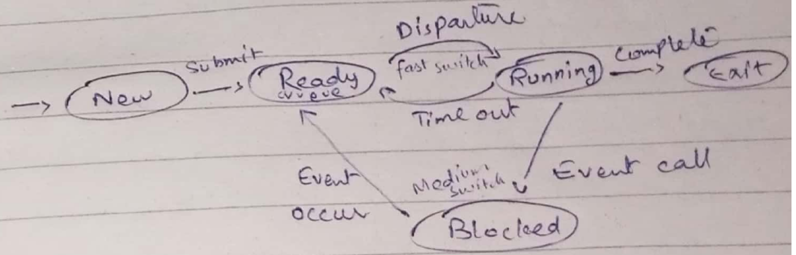
To like round robbin.



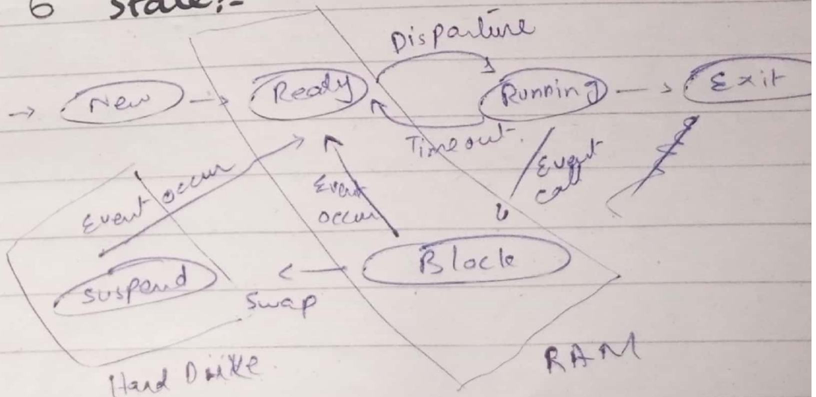


It is not compelete bcz if any process deman any resource so, process not com

### 5 stalé:-



### 6 stalé:-



If same time event occur in block and suspend. So higher piority is block state.