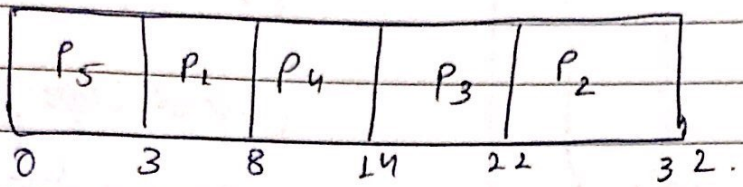
Page No.:
Date:
YOUVA

Shortest Job First (SJF)

P_1	5
P_2	10
P_3	8
P_4	6
P_5	3



$P_1 wt = 3$

$P_2 wt = 22$

$P_3 wt = 14$

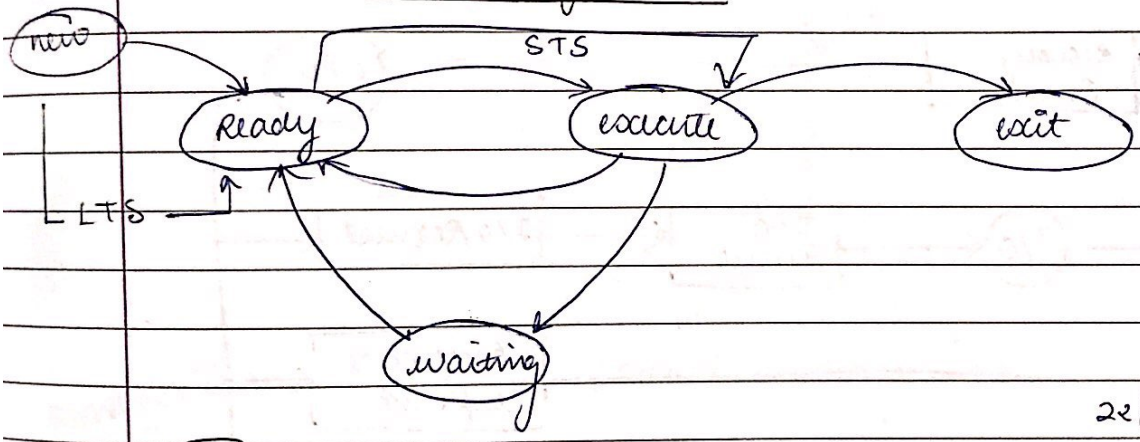
$P_4 wt = 8$

$P_5 wt = 0$

47

$P_{avwt} = 47/5 = 9.6$

Process State Diagram :-

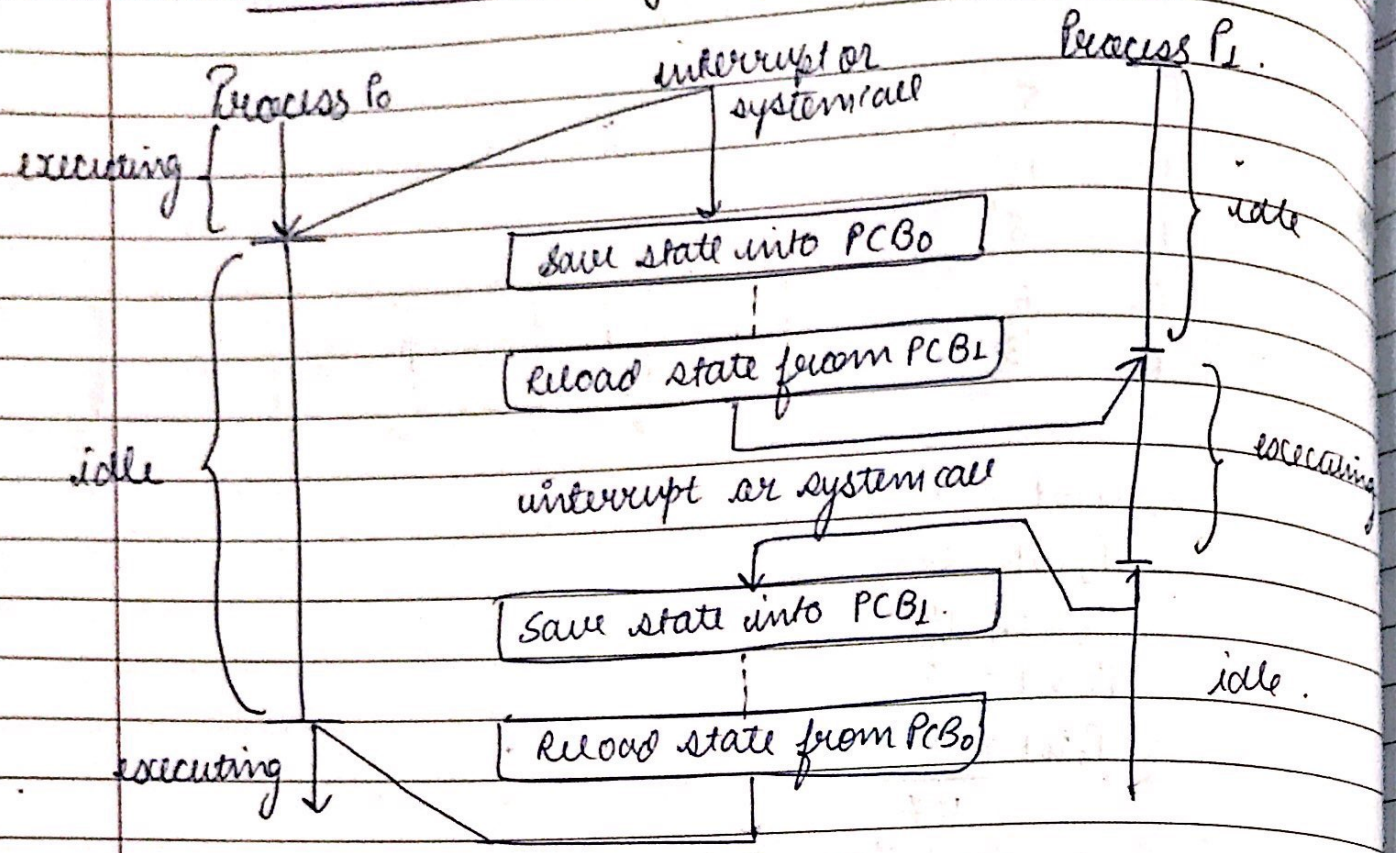


22/7/19

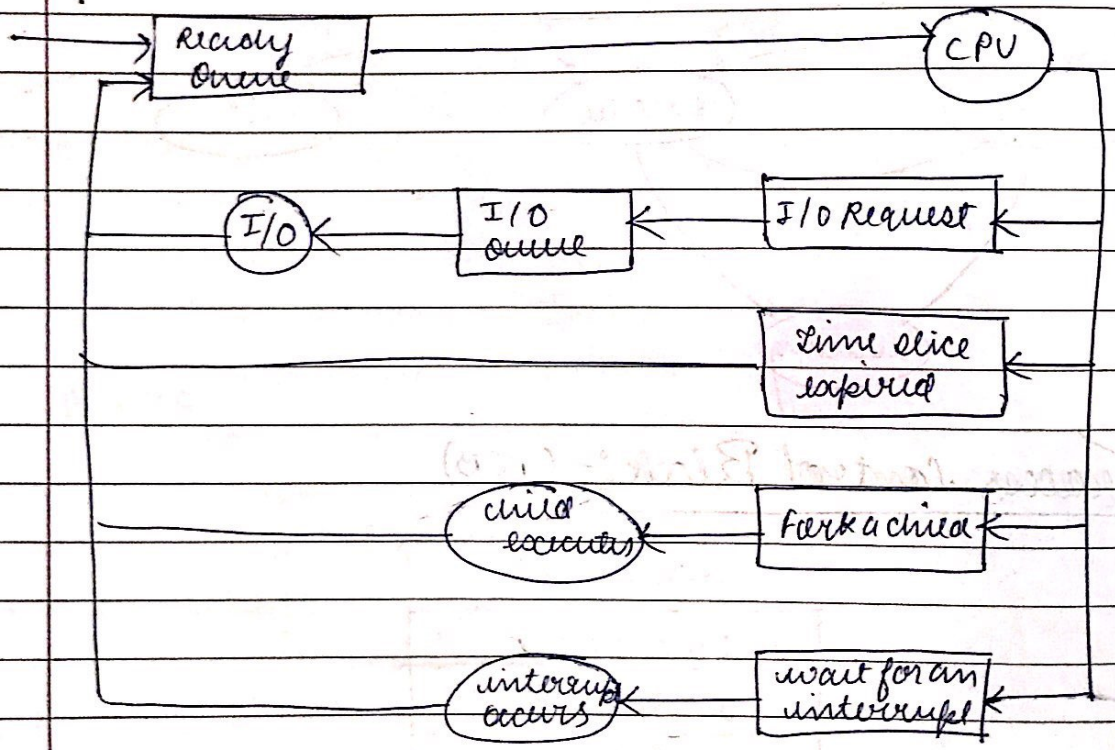
Process Control Block :- (PCB)

Process state
Process NO.
Program counter
Registers
Memory limits
List of open files
:

Context switching :-

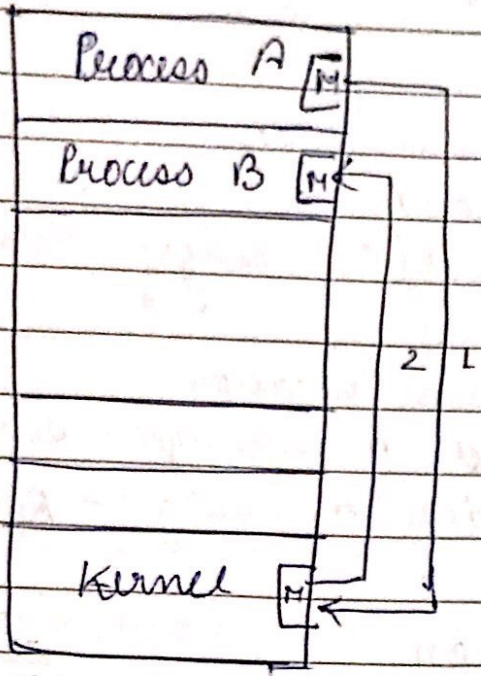


CPU switches from process to process.

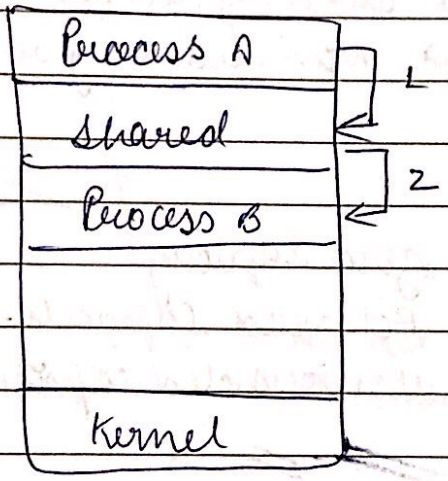


Queuing Diagrams representations of Process scheduling.

Interprocess Communicator (IPC)



(i) Message Passing



(ii) Shared memory

8 marks *imp

① Shared Memory

a.) Producer - Consumer.

b.) Buffer

(i) Bounded

(ii) Unbounded

Date: YOUVA

(2) Message Passing

a.) Naming

(i) Direct communication

→ send(P, message) - send a message to process P.

→ recieve(Q, message) - Recieve a message from process Q.

(ii) Indirect communication

→ send(A, message) - Send a message to mailbox A

→ recieve(A, message) - Recieve a message from mailbox A.

b.) Synchronization

→ Blocking send

→ Nonblocking send

→ Blocking recieve

→ Non blocking recieve.

c.) Buffering

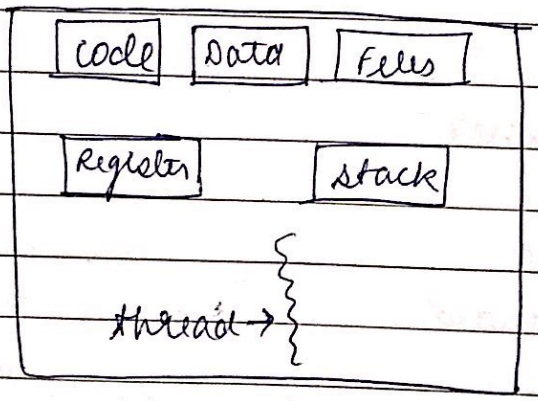
→ zero capacity

→ Bounded capacity.

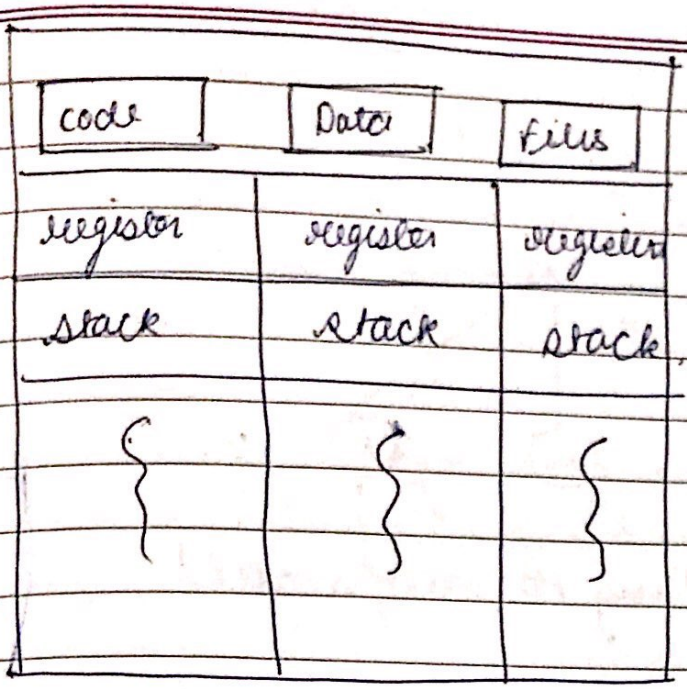
→ unbounded capacity.

* Threads:

→ Smallest unit of process.

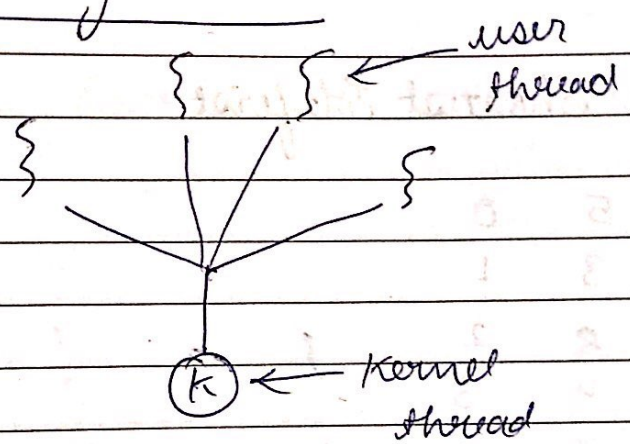


Single threaded process

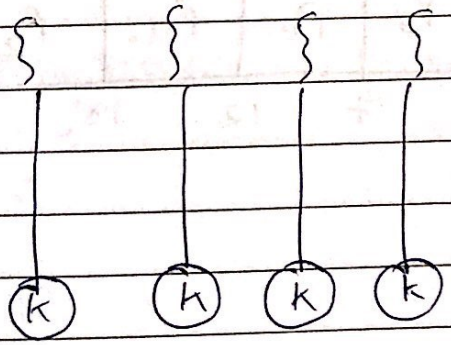


Multithreaded Process

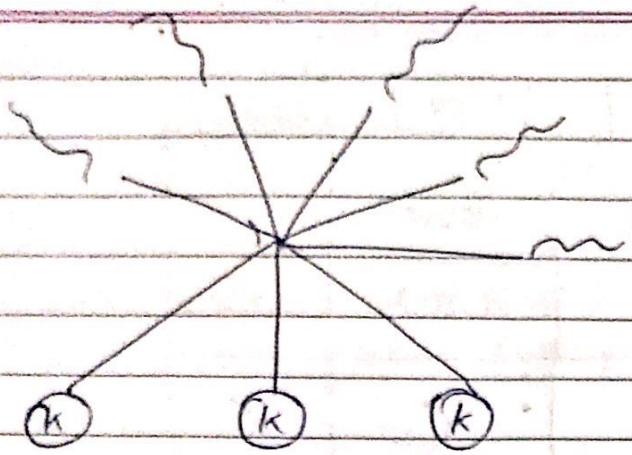
Multithreading Models :



(Many to one model)



(one to one model)

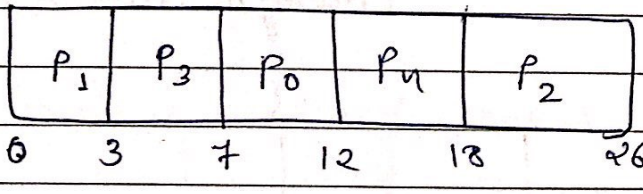


(Many to many model)

* Scheduling Algorithms:

1. FCFS
2. SJF - Shortest Job first.

P_0	5	0
P_1	3	1
P_2	8	2
P_3	4	3
P_4	6	4



$P_{0_{wt}} - 7$

$P_1 wt - 0$

$P_2 wt = 0$

$P_2 wt - 18$

$P_3 wt - 3.$

$P_4 wt - 12$

(3) SRTF :-

	Arrival	Burst
P_0	0	8
P_1	1	4
P_2	2	9
P_3	3	5

P_0	P_1	P_3	P_0	P_2
0	1	5	10	17	26

waiting time :-

$P_0 \rightarrow 9$

$P_1 \rightarrow 0$

$P_2 \rightarrow 15$

$P_3 \rightarrow 2$

turnaround -

$P_0 \rightarrow 17$

$P_1 \rightarrow 5-1$

$P_2 \rightarrow 26$

$P_3 \rightarrow 10$

8.	P_0	5	0
	P_1	3	1
	P_2	8	2
	P_3	4	3
	P_4	6	4

P_0	P_1	P_0	P_3	P_4	P_2
0	1	4	8	12	18	26

waiting time -

$P_0 \rightarrow 3$

$P_1 \rightarrow 0$

$P_2 \rightarrow 16$

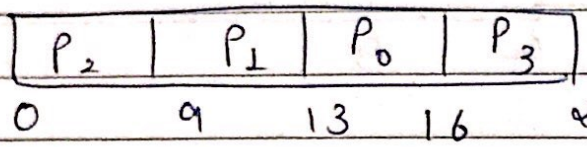
$P_3 \rightarrow 8$

turn-around time -

Page No.:
Date:

(4) Priority Scheduling –

	Priority	Burst
8-) P_0	3	8
P_1	2	4
P_2	1	9
P_3	4	5

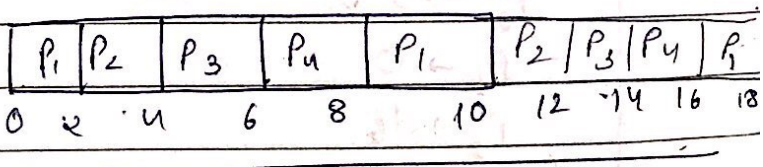


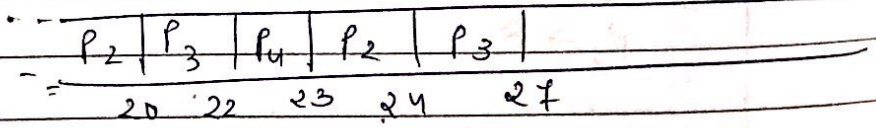
★ Starvation – Starvation is a problem in which the lowest priority process gets unserved.
For this problem we always add 1 with the ~~p~~ every priority.

(5) Round Robin Scheduling:-

quantum = 2ms.
(executing)

	Burst time
P_1	6
P_2	7
P_3	9
P_4	5





Page No.:
Date:

waiting time

P_1 wt → 12

P_2 wt → 18

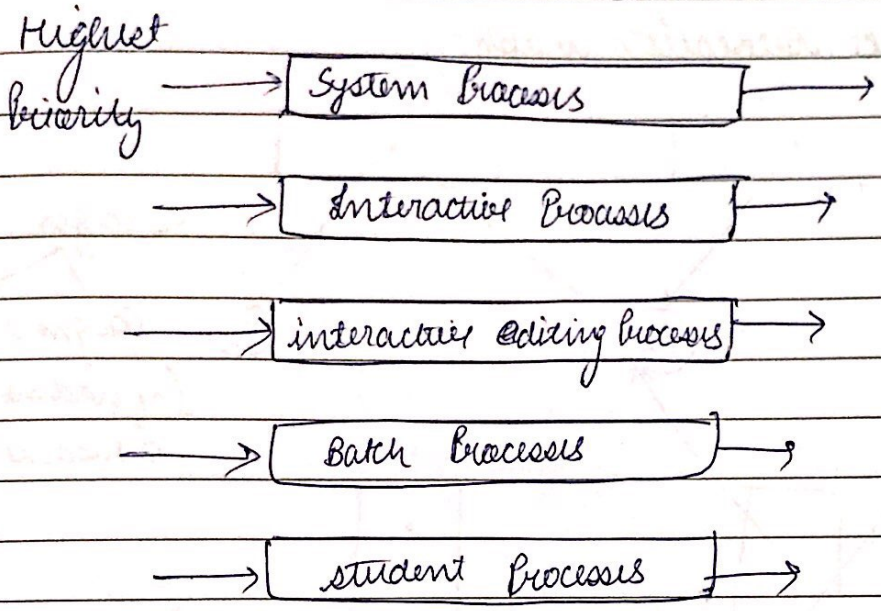
P_3 wt → 28

P_4 wt → 18 = 65/4

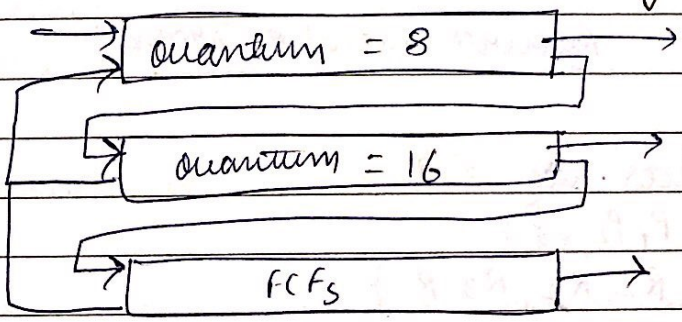
= 16.25

8 marks

⑥ Multilevel Queue Scheduling:



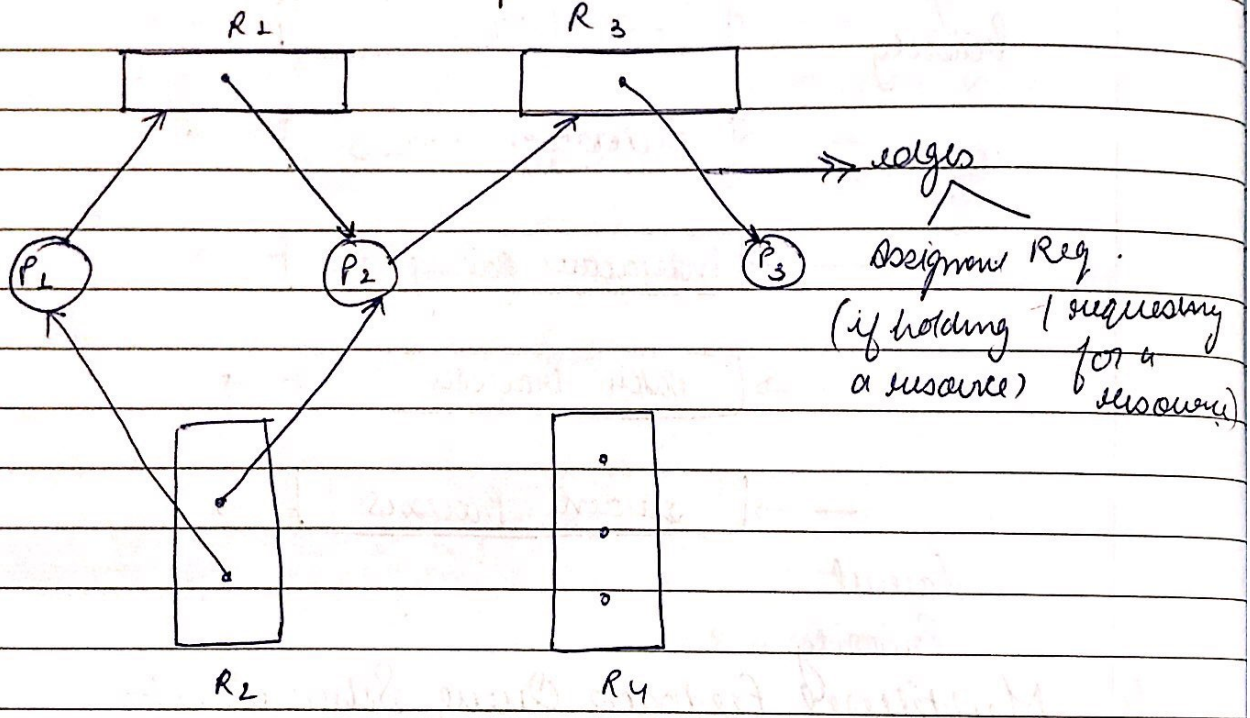
⑦ Multilevel feedback Queue Scheduling:-



Deadlock :- (4 marks) -

* Necessary condition :-
→ Mutual exclusion - At least one resource should be in non-shareable mode.
→ Hold & wait - One system is holding a resource & demanding for other.
→ No preemption - If no resource can be preempt.
→ circular wait -

Resource Allocation Graph -



Resource allocation Graph.

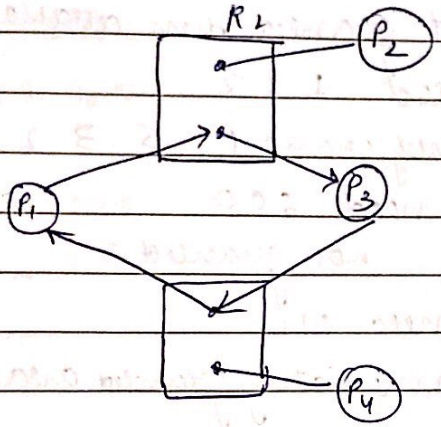
- The sets of P, R, E:
→ P = { P1, P2, P3 }
→ R = { R1, R2, R3, R4 }
→ E = { P1 → R1, P2 → R3, R1 → P2, R2 → P2, R2 → P1, R3 → P3 }

- Resource Instance

→ one instance of Resource R1

→ Two Instance of Resource R_2
→ One Instance of Resource R_3
→ Three Instance of Resource R_4

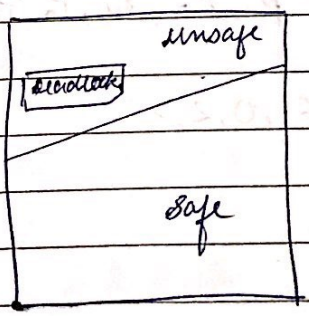
• Process state
→
→
→



* Deadlock Prevention :-
(i) Mutual exclusion
(ii) Hold & wait
(iii) No Preemption
(iv) circular wait.

* Deadlock Avoidance :

(i) Safe, unsafe or deadlock



* Banker's Algorithm :-

	Allocation	Max(need)	Available	Actual Need
P_0	0 1 0	7 5 3	3 3 2	7 4 3
P_1	2 0 0	3 2 2		1 2 2
P_2	3 0 2	9 0 2		6 0 0
P_3	2 1 1	2 2 2		0 1 1
P_4	0 0 2	4 3 3		4 3 1

For P_0 we need 7 4 3
Not granted then available 3 3 2.

For P_1 we need 1 2 2.
immediately granted 5 3 2

For P_2 we need 6 0 0
not granted.

For P_3 we need 0 1 1
immediately granted avail 7 4 3

For P_4 we need 4 3 1
immediately granted available 7 4 5

For P_0 we need 7 4 3
immediately granted 7 5 5.

For P_2 we need 1 2 2
immediately granted 10 5 7

$< P_1, P_3, P_4, P_0, P_2 >$

$P_1 <1, 0, 2>$

Deadlock Detection :-

i) single Instance of each resource type.
ii) Several Instance of a resource type.
iii)

Recovery from Deadlock :-

(i) Process termination
→ Abort all deadlock process.
→ Abort one process at a time until the deadlock cycle is eliminated.

(ii) Resource Preemption
→ Selecting a victim.
→ Rollback.
→ starvation.

	Allocation				Max				Available				Actual/Max			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	1	2	0	0	1	2	1	5	2	0	0	0	0	0
P_1	1	0	0	0	1	7	5	0					0	7	5	0
P_2	1	3	5	4	2	3	5	6					1	0	0	2
P_3	0	6	3	2	0	6	5	2					0	0	2	0
P_4	0	0	1	4	0	6	5	6					0	6	4	2

=) if a request process P_1 arrives for (0, 4, 2, 0) can the request be granted immediately.

ans yes.

= For P_0 we need 0000
granted 1520

For P_1 we need 0750
not granted 1520.

= For P_2 we need 1002.
granted 2886

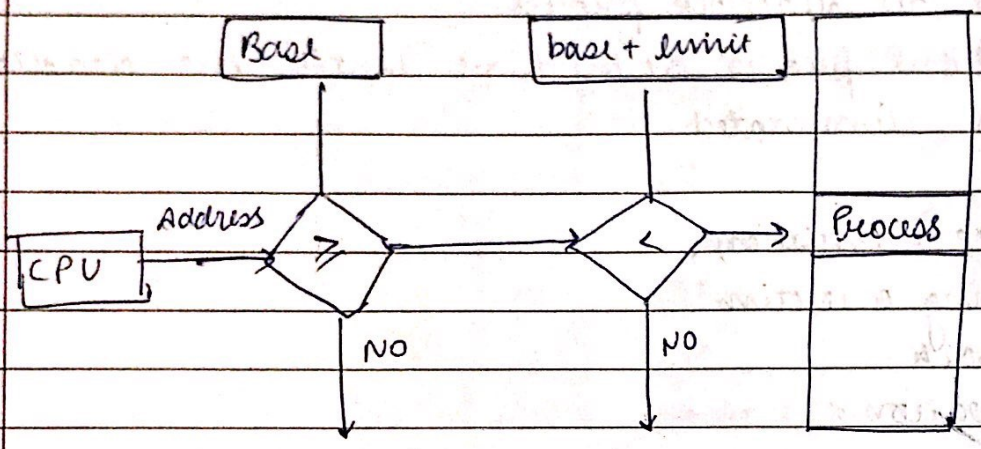
= For P_3 we need 0020
granted 21118

= For P_4 we need 0642
immediately granted
2 14 12 12

For P_L we need 0750
granted 3 14 12 12

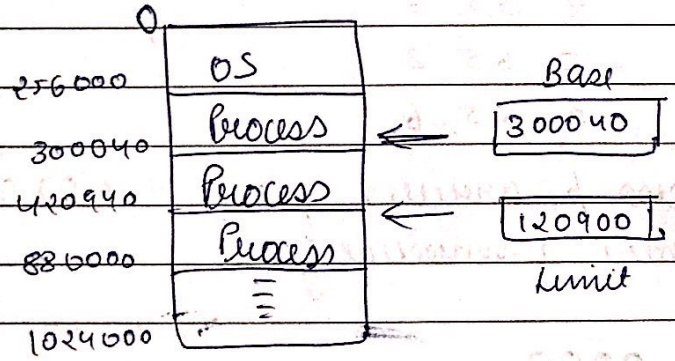
$< P_0, P_2, P_3, P_4, P_1 >$

Basic Hardware:



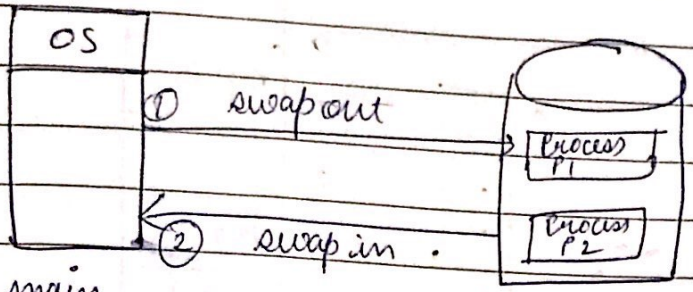
Trap to operating system
monitor - addressing error.

② H/W address protection with Base & limit register.



① A base & limit Register define base & limit register.

* Swapping :-



swapping of two process

* Memory Allocation :-

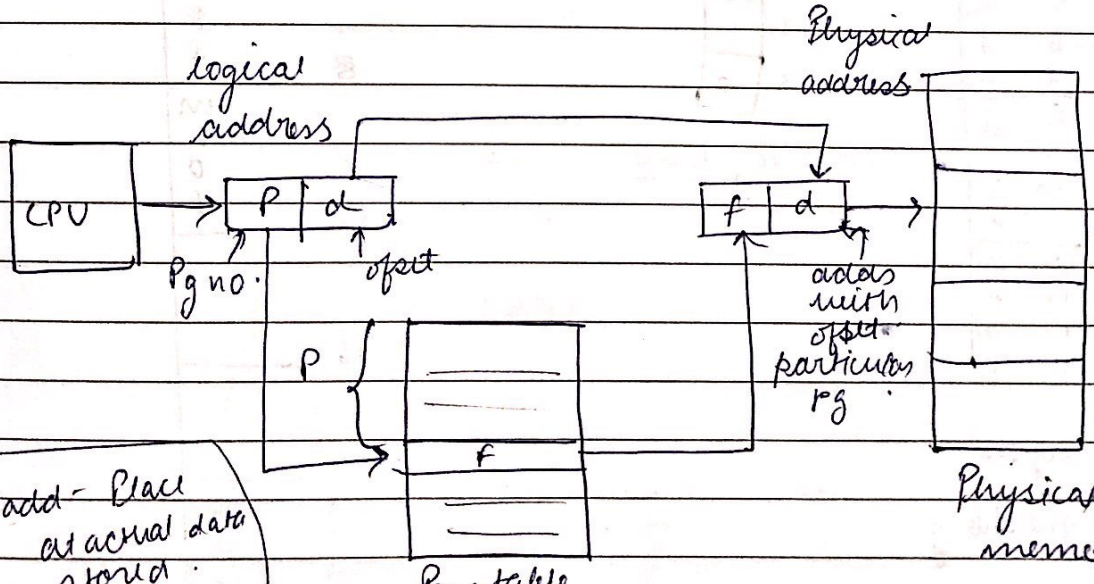
→ first fit
→ best fit
→ worst fit.

* Fragmentation

→ external fragmentation
→ internal fragmentation (fixed sized memory blocks)

* Paging :

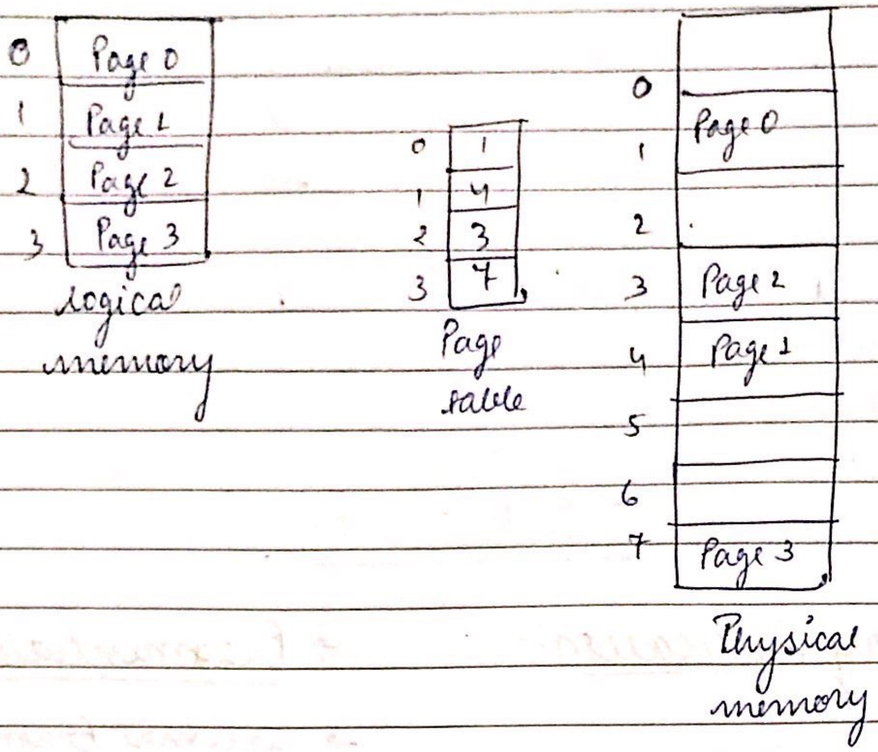
(used to avoid external fragmentation)



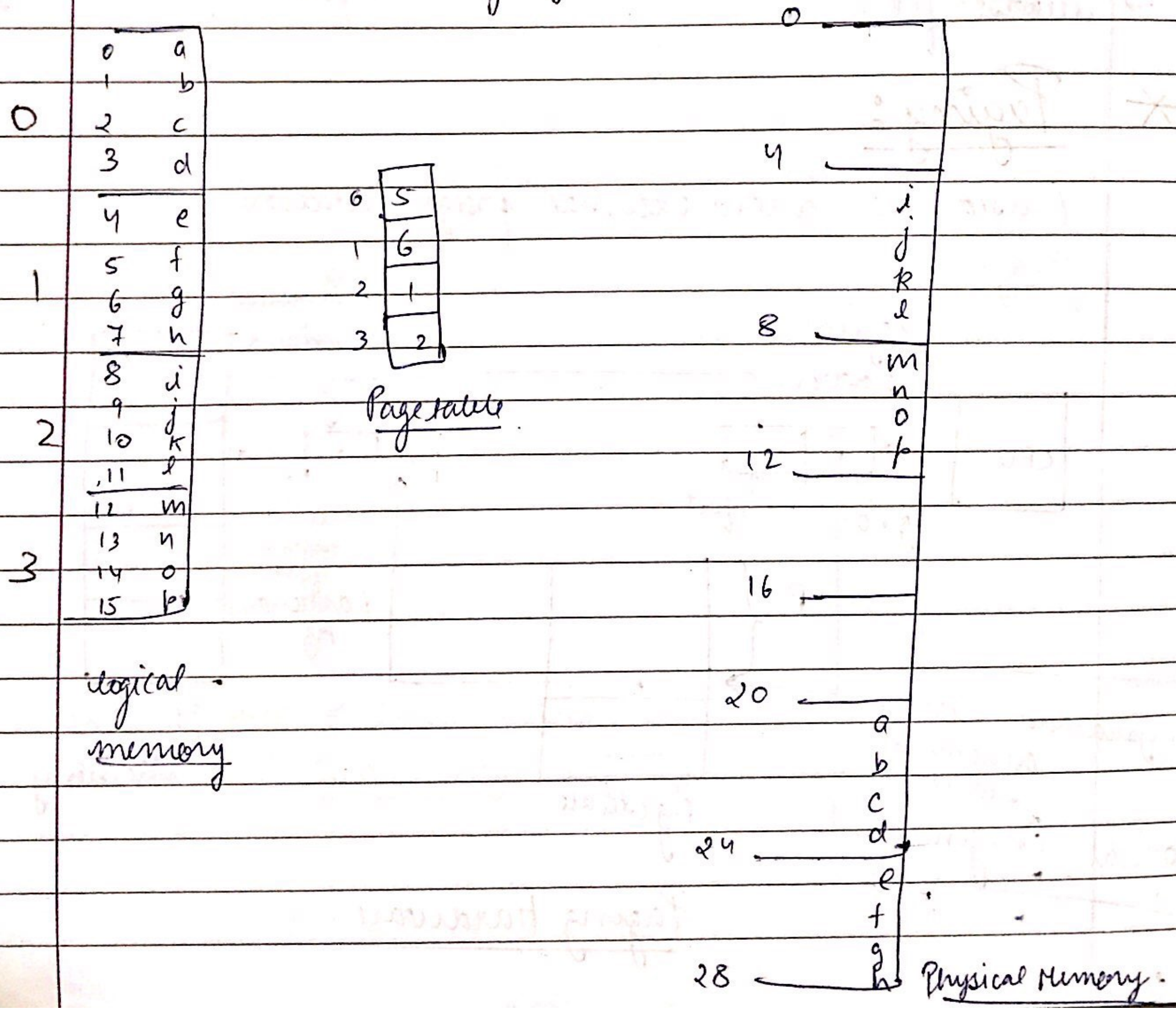
Physicaladd - Place at actual data stored.
logical - CPU generates

Paging Hardware

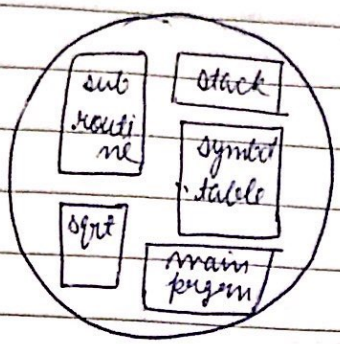
Date:



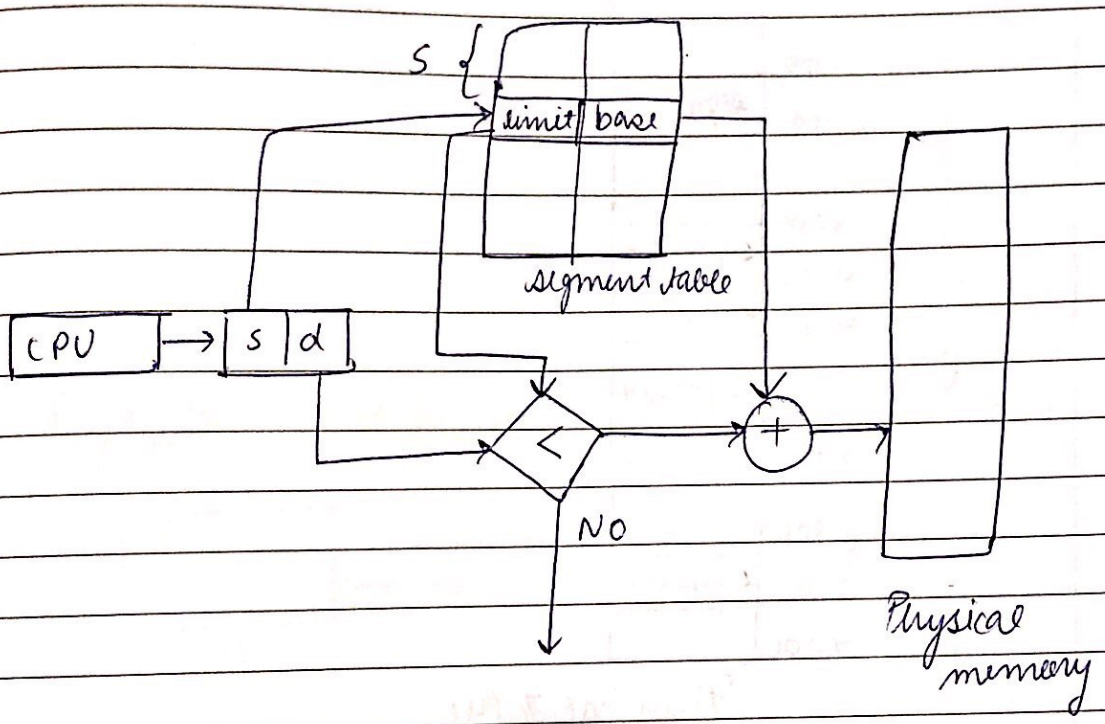
Paging Model



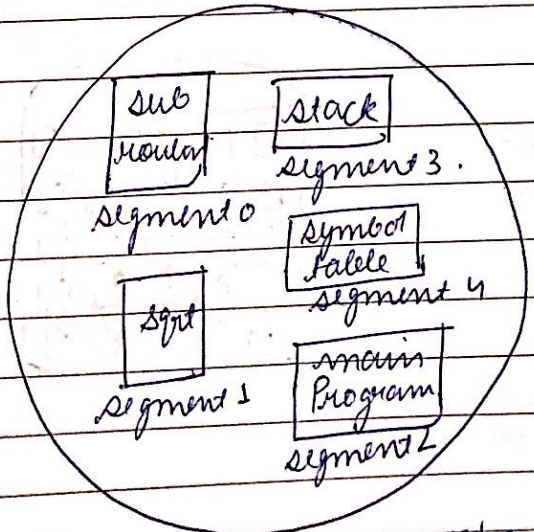
Segmentation :-



logical address.



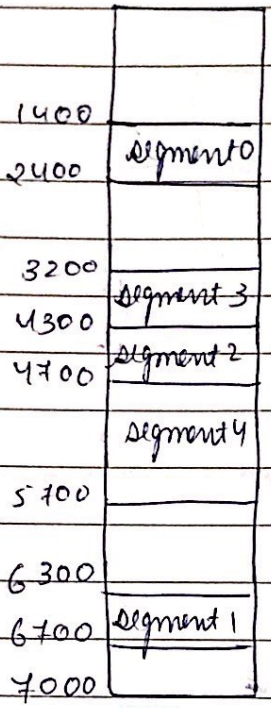
Segmentation Hardware:



logical address space

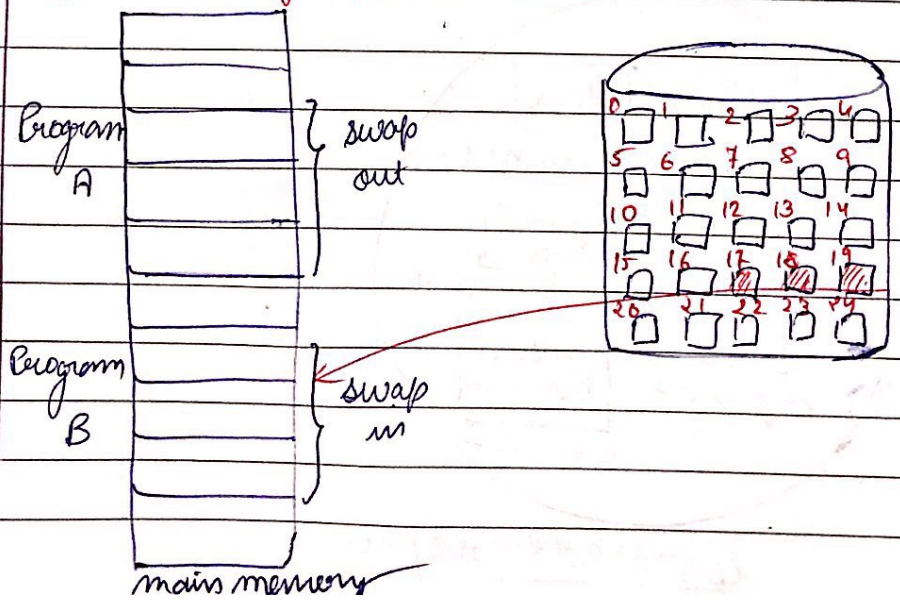
limit	base
1000	1400
400	6300
400	4300
1100	3200
1000	4700

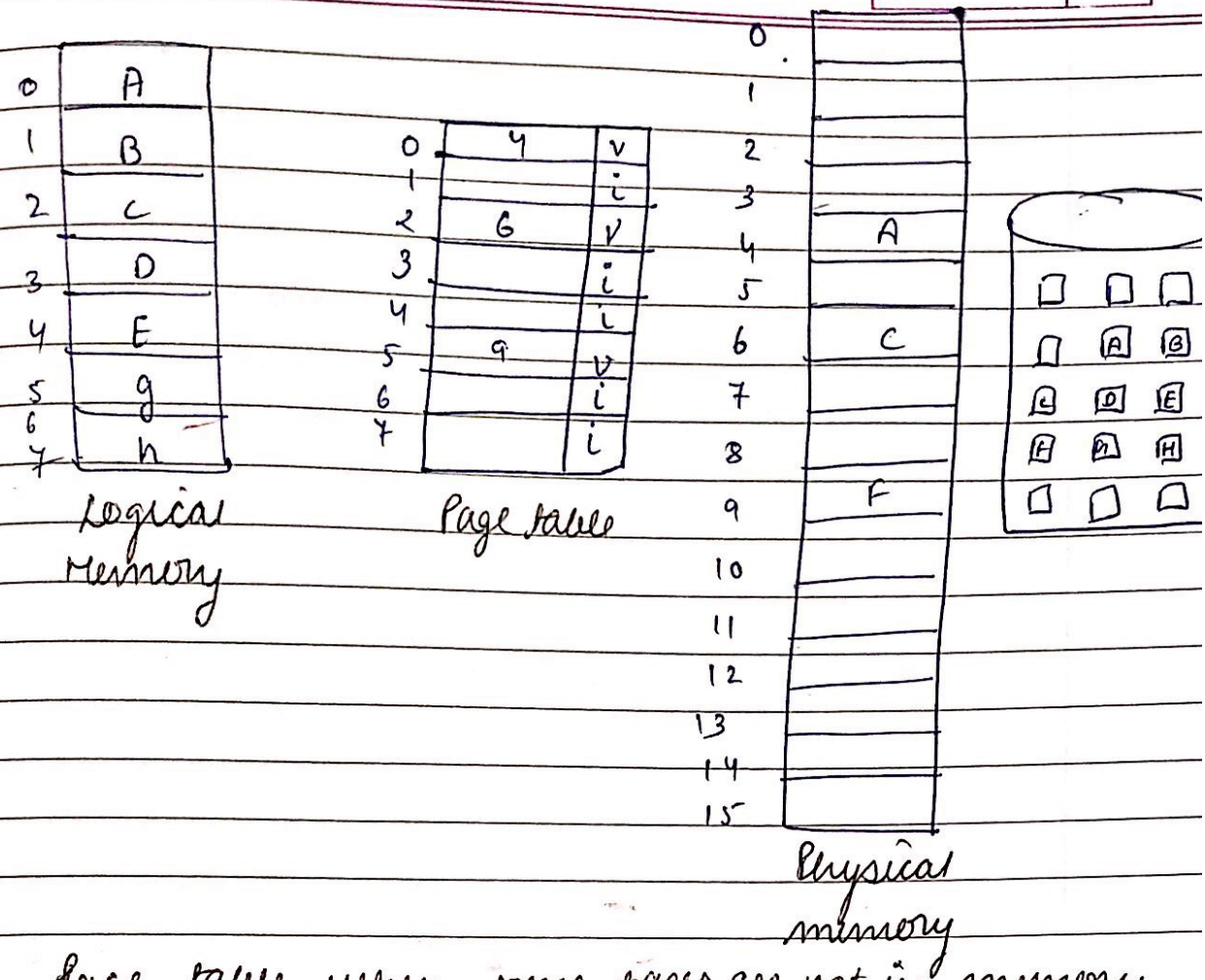
Segment table



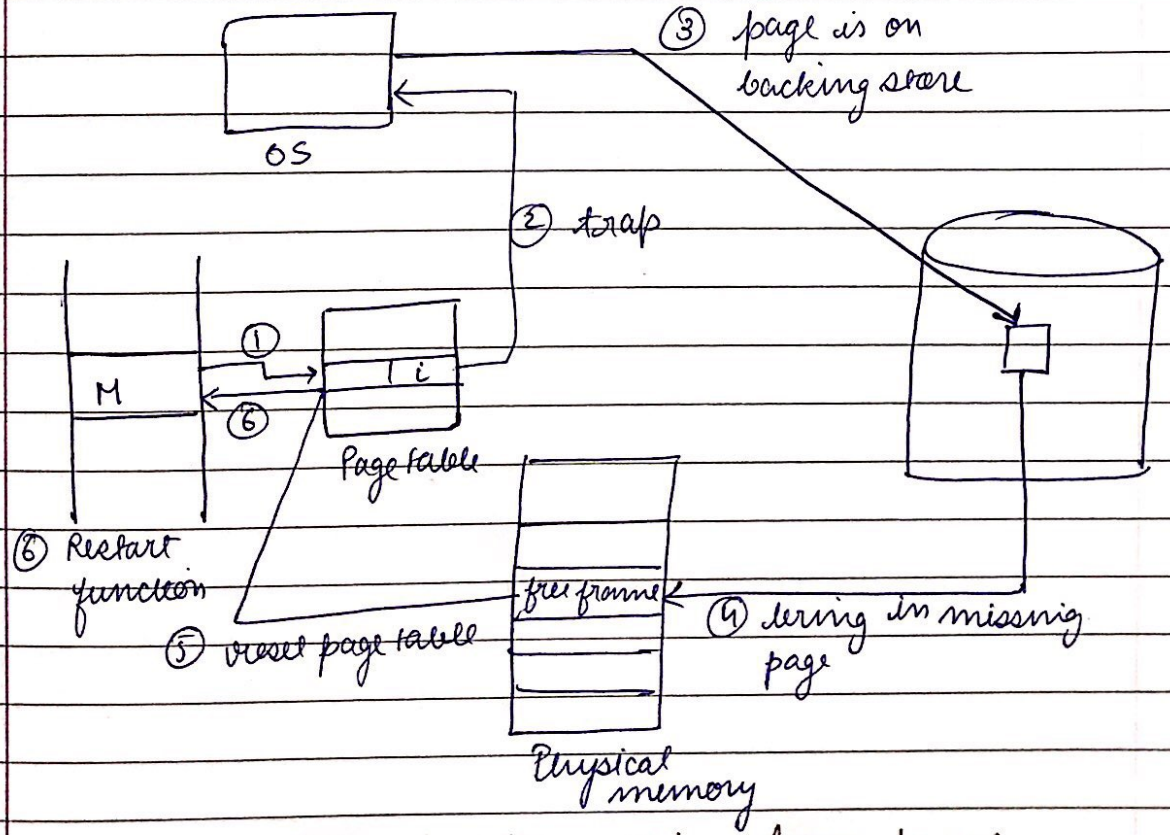
Physical Table

* Demand Paging :-





Page table when some pages are not in memory.

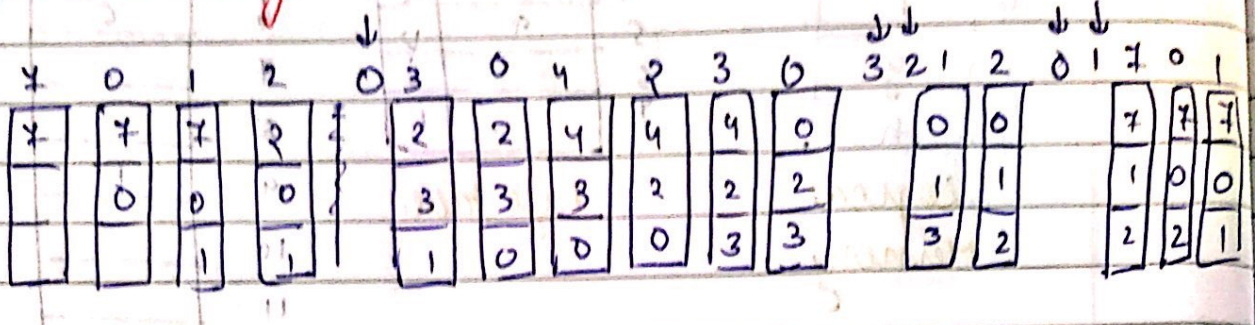


steps in handling page fault

* Page Replacement algorithm :-

(1.) First in First out :

Reference string = 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1



Page hit = 5

Page fault = 15

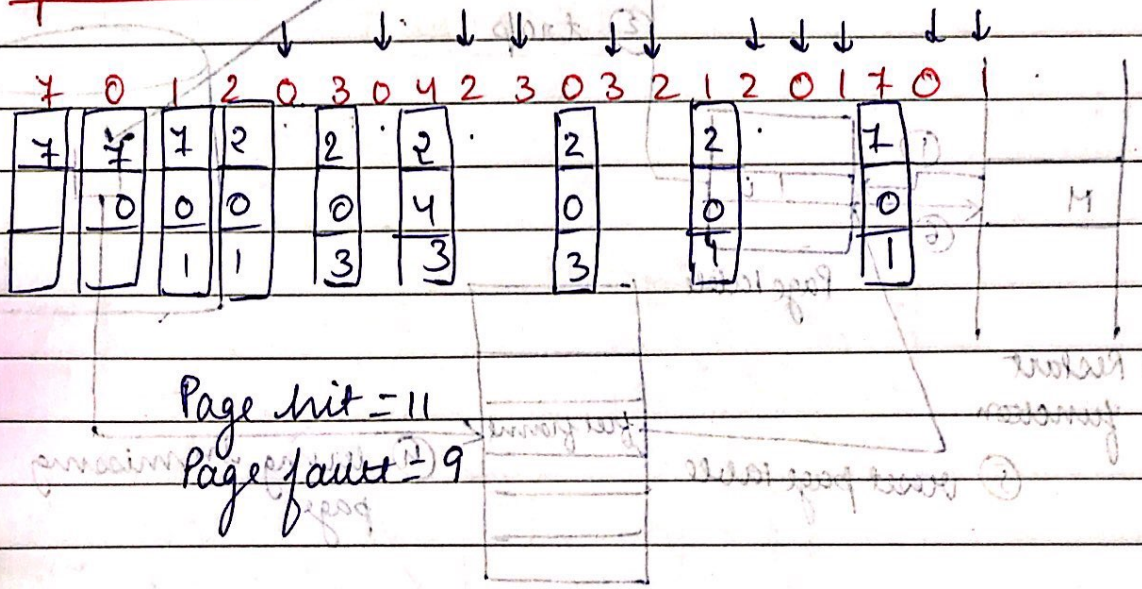
=> Belady Anomaly

Belady Anomaly -

If we increase the number of frames it will also increase the page fault.

This problem occurs only in FIFO.

(2.) Optimal :-

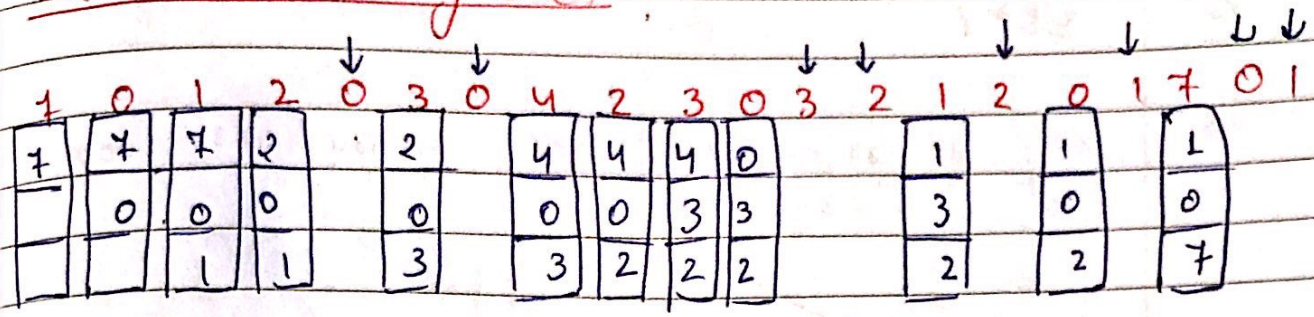


Page hit = 11

Page fault = 9

Page No.:
Date:

(2) LRU (Least Recently used)

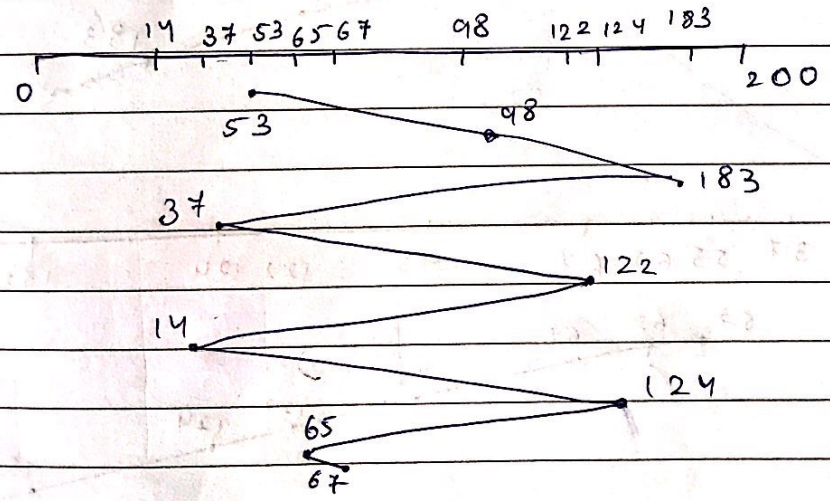


* Disc Scheduling:-

1. FCFS
2. SSTF
3. SCAN
4. C-SCAN
5. LOOK
6. C-LOOK-

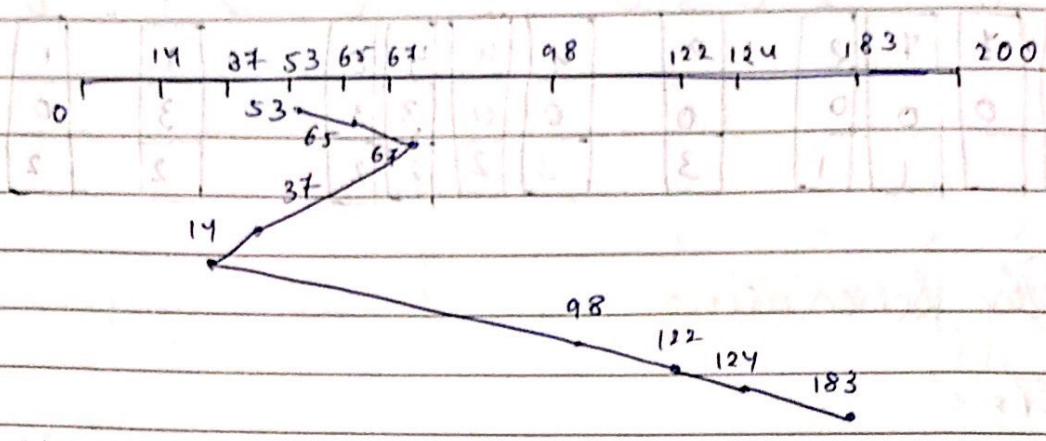
=> 98, 183, 37, 122, 14, 124, 65, 67

=> Head starts at -53.



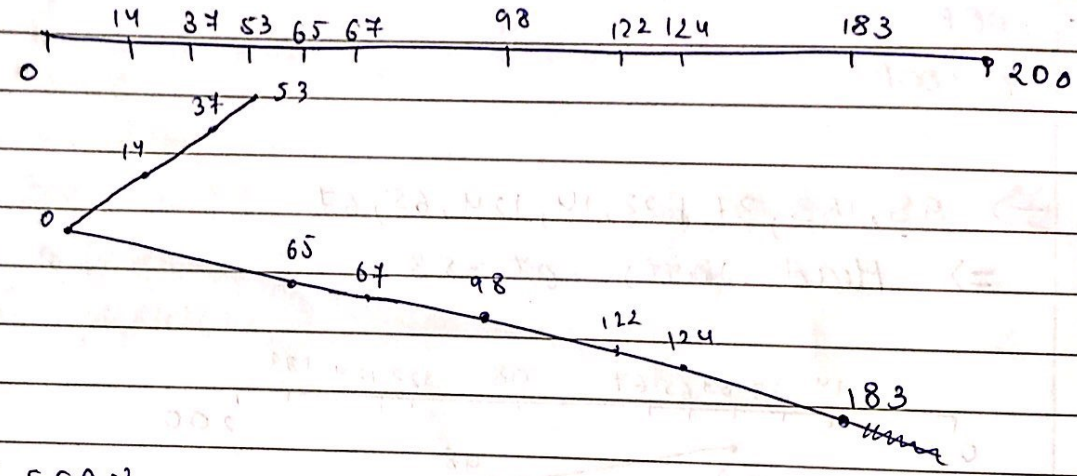
(98-53) + (183-98) + (183-37) + (122-37) + (122-14)
+ (124-14) + (124-65) + (67-65)
= 640.

(2) SSTF.

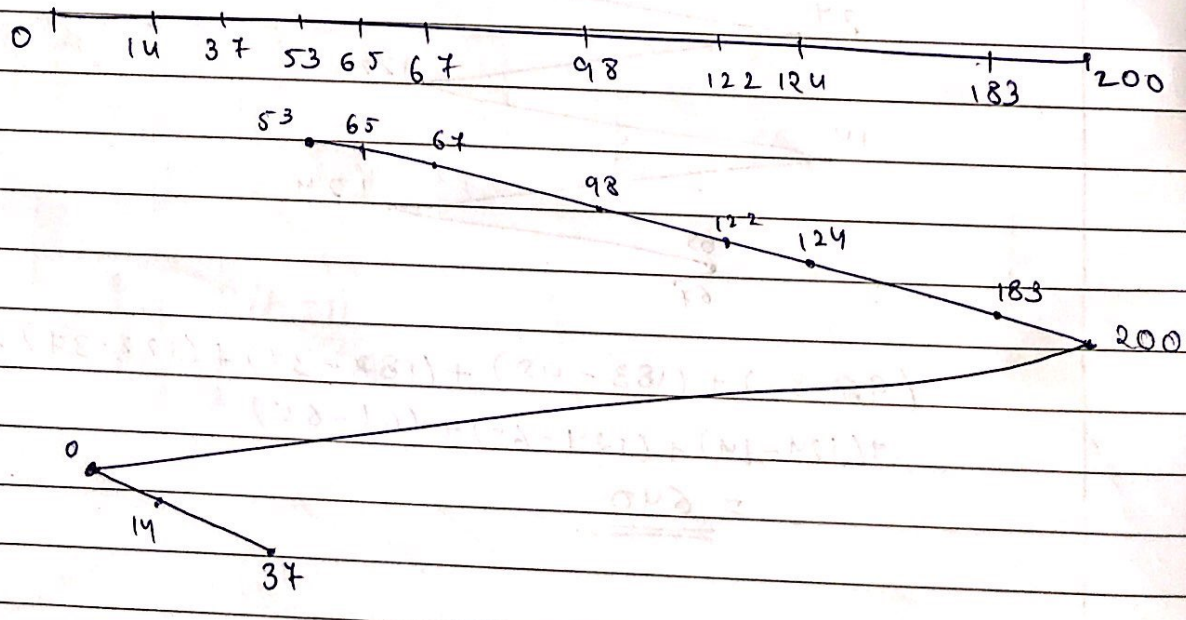


No. of moves = 236

(3) SCAN

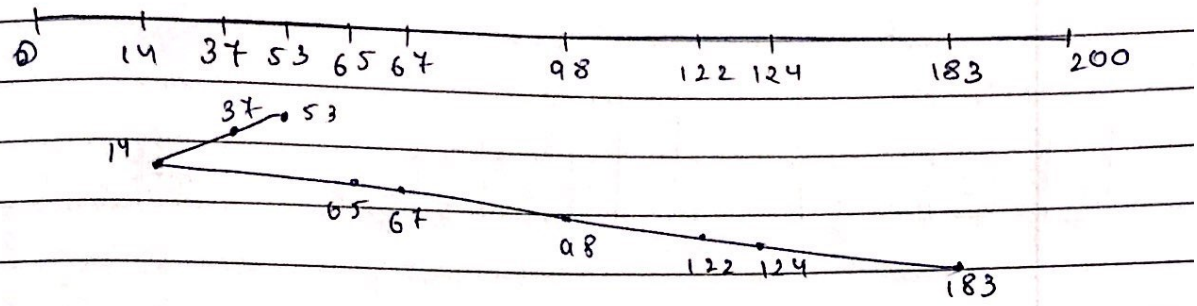


(4) C-SCAN



Page No.:
Date:

(5) LOOK:



(6) ~~SCAN~~ : C-LOOK :