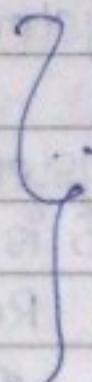


Cosmos

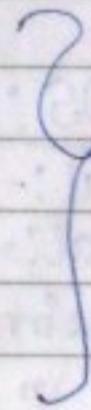
Text Books :-
 ① Galvin
 ② Modern OS by Tannerbaum
 ③ OS by William Stallings

Schedule:-

1. Introduction & background
2. Process Management
 - process concept
 - CPU Scheduling
 - Synchronisation
 - Concurrent Programming
 - Deadlocks
 - Threads
3. Memory Management
 - RAM chip Implementation
 - Loading & Linking
 - Address Binding
 - Paging
 - multi-level paging
 - inverted paging
 - Segmentation
 - Segmented Paging
 - Virtual Memory
4. File & Device Management
5. Protection & Security



→ 40%

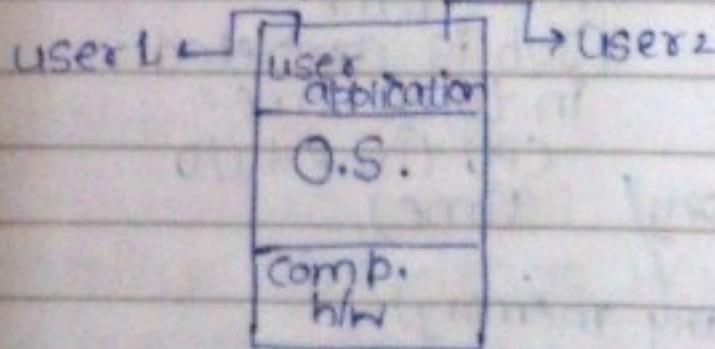


→ 40%

Introduction & Background

- What is an Operating System :-

It is an interface b/w user & computer hardware.



main()
{ int x;

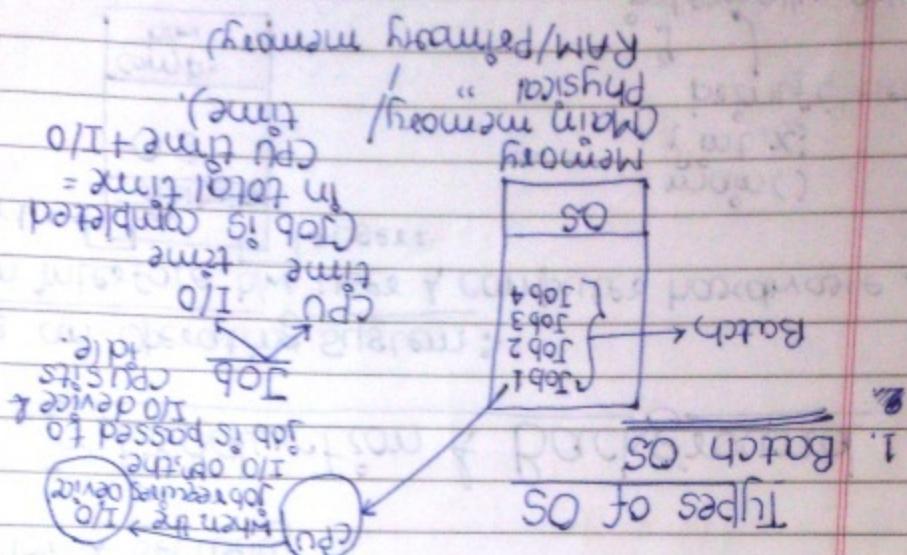
printf("Hello World")

internally calls write system call in order to communicate with monitor.

System Call:-

It is a request made by the user program in order to

take in order to execute the job, the job must reside in the main memory.



Goals of OS:-

- Primary Goal:- convenience (ease of use)
- Secondary Goal:- efficiency
- Windows primary goal is convenience.
- Linux " " efficiency.

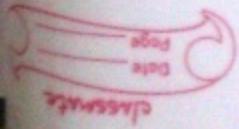
OS acts as government which manages every thing inside a computer.

Operating system can also be called as resource allocator. OS is responsible for resource allocation.

Kinds of system calls.

For diff. kind of services, there are diff. get any kind of service from OS.

COSMOS



Cosmos

In Batch OS, if a job requires I/O opn. In b/w, then the CPU will sit idle till that job performs its I/O opn. & comes back to the CPU, i.e. in b/w the CPU won't be allocated to other job.

classmate

Date _____

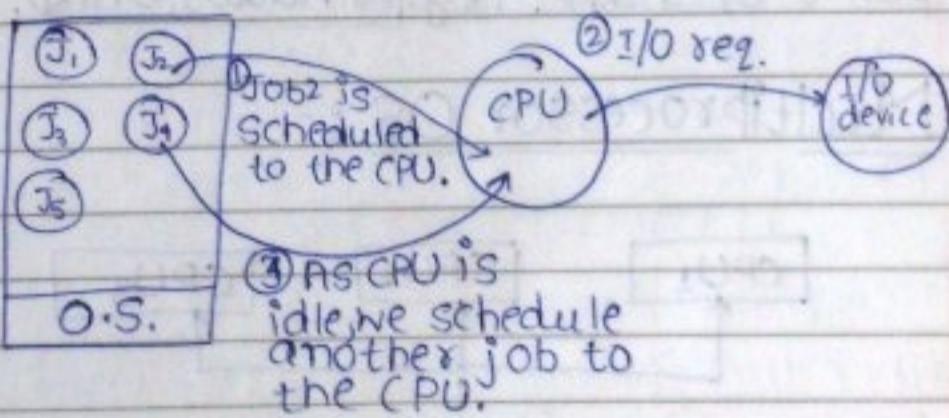
Page _____

- ★ The primary responsibility is to place the job into CPU.
- ★ The other job is scheduled (given to CPU) only when the job is completed completely.
Disadvantages:- CPU utilization is less, so throughput of the system will decrease.

Throughput:- The no. of jobs executed / unit time is called throughput of the system.

e.g. [IBM OS/2]

2. Multiprogramming OS

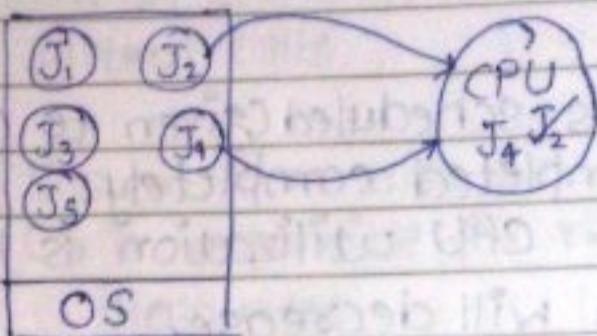


Advantage:- increased CPU utilization & hence increased throughput of the system.

3. Multitasking

The multitasking OS is an extension to Multiprogramming OS. The jobs will be executed on the CPU in time sharing mode.

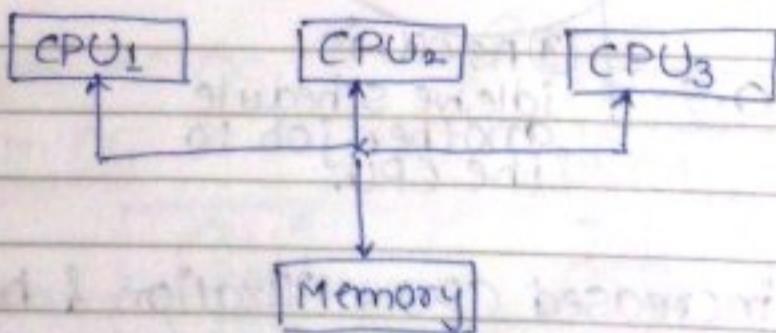
Cosmos



This takes place like:-

- For a particular amount of time J₁ is executed.
- When the time expires, Job J₂ preempts J₁ & J₂ is scheduled to CPU.
- ★ Only one job is executed on the CPU at any point of time. (e.g. Windows, Unix).

Multiprocessor OS :-



- ★ One system with multiple processors.
- OS is required to support h/w.

Advantages :-

1. Increased Throughput of the System.
2. Reliability: If one CPU fails, the other CPUs will take care that entire system doesn't crashes.

It is also called Fault Tolerance System.

Cosmos

classmate

Date _____

Page _____

3. It is economical.

e.g. buying 1 system with n CPUs < buying individual systems with 1 CPU each.

Real Time OS :-

The systems which are strictly time bound

if an instrn takes n ns to execute, it should execute in not more than n ns.

Hard Real Time

e.g. missile systems,
satellite system
There should not be
any delay at all.
e.g. no delay is accepted
in missile launch.

★ e.g. RTOS

Soft Real Time

e.g. Banking System
[minor delays are
accepted.] e.g., some
delay in updation of
balance.

Process Management

Process Concept :-

Defn. :- program under execution (process).

- ① The program should reside in the main memory
 - ② It has occupied the CPU to execute the instrn.
- Process has attributes, states & is able to perform various operations.

Attributes:-

- Process id
- Process state
- Program Counter
- Priority
- General purpose reg.
- List of Open files
- List of " devices
- Protection info.

States:-

- New
- Ready
- Running
- Wait (or) Block
- Termination (or)
- Completion
- Suspend Ready
- Suspend Wait (or)
- Suspend Block

Operations :-

- creation
- scheduling
- executing
- killing / termination

Cosmos

classmate

Date _____
Page _____

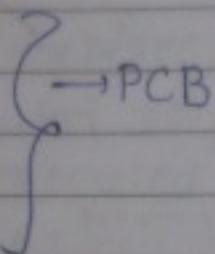
Process id: It is a unique identification no. which is assigned by OS during process creation.

Process State: It contains the current state info. where the process is residing.

Program Counter: The PC contains the address of next instⁿ to be executed.

Priority: Priority is a parameter which is assigned by the OS at the time of classes creation.

Pid	P.State
PC	Priority
General Purpose Reg.	List of Open files
List of open devices	Protection info



* every process has its own PCB.

Process State Diagram:-

- New:- the process is under creation, & when the process is created it moves to Ready state.
under creation means it is being shifted to the main memory.
- Ready:- in ready states, there is a no. of processes, then we select one process & we schedule it to the running state.
- Running:- in this state, CPU executes the instⁿ of the running process. In this state, there is only one process. If the process requires any I/O operⁿ, the process will move to Wait or block state.
- Wait/Block state:- process performs I/O operⁿ. When I/O operⁿ completes, the process

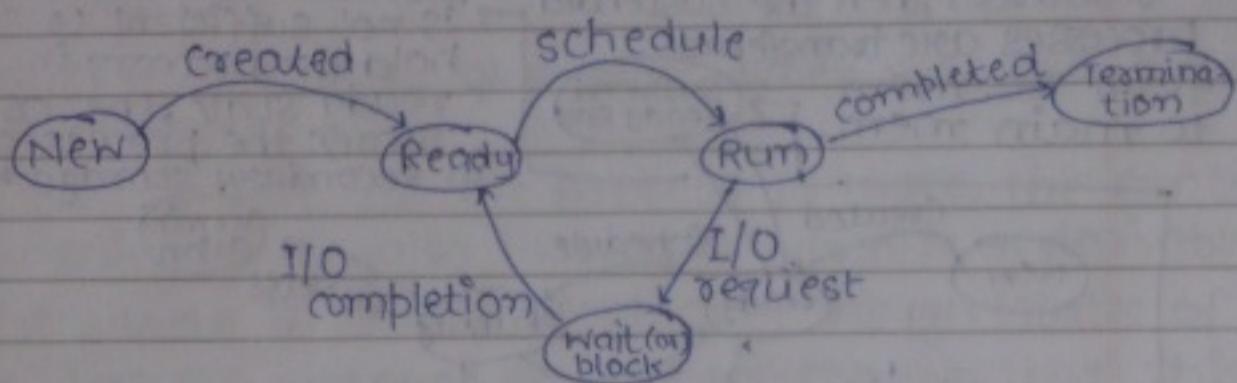
Cosmos

classmate

Date _____
Page _____

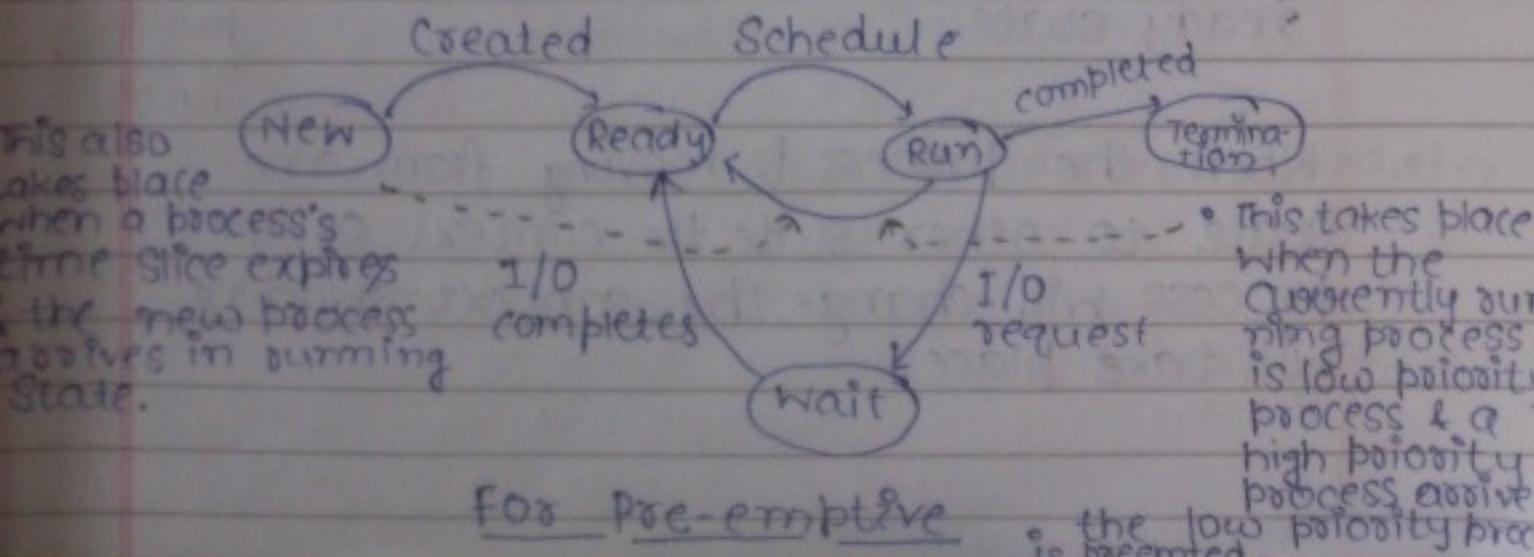
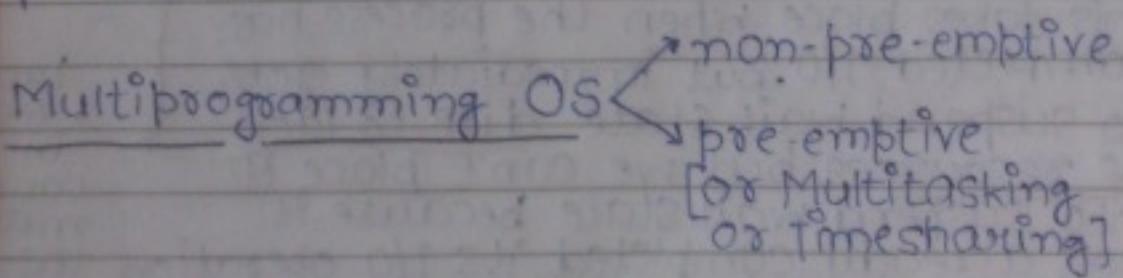
goes to ready state.

- If the running process completes process moves to termination or completion state.



For Non-preemptive

- In wait state also, we can have multiple no. of processes.



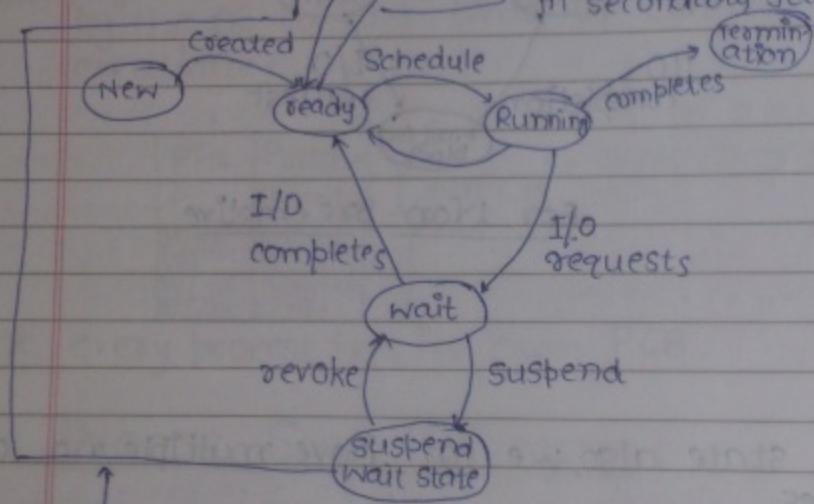
Cosmos

- When the process is in the ready, running or wait state, it means process is in the main memory.

* & When the comp.

or mem. has sufficient resources, then the suspended processes are transferred from sec. mem. to main mem.

When the main memory is not sufficient to hold the processes in ready state, the OS stores the processes in secondary storage *



this takes place when the process has completed I/O, but immediately goes to suspend wait state due to lack of resources, but we can't place it into suspend wait state because it has already completed its I/O operation & must be placed into suspend ready state.

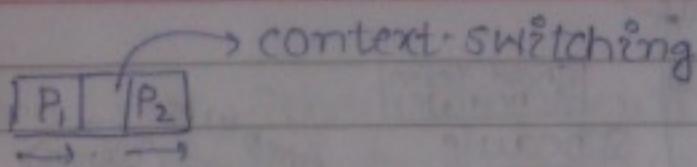
{ lack of resources in this context means that the RAM is full, + process can't be placed in RAM.

- When the process is moving from one state to other state, the context of the process will change, the context-switching will take place.

Cosmos

classmate

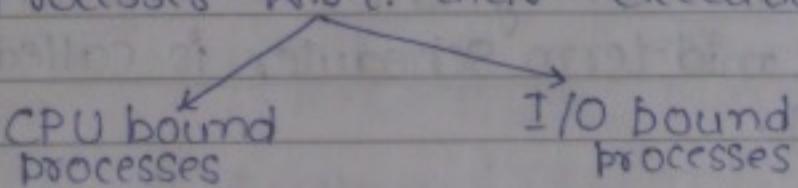
Date _____
Page _____



Saving the context of one process & loading the context of other process is called as context switching.

- ★ Context-switching takes place when we have atleast two processes.
- ★ If the context of the process is more, the context switching time also increases which is undesirable.
- ★ When there is one process only, though states of the process changes, so context changes, but this is not context-switching.

Processes w.r.t. their execution time



- The processes which require more CPU time are called CPU bound processes.
- These processes will spend more time in running state.
- The processes which require more I/O time are called as I/O bound processes.
- These processes will spend more time in wait state.

★ In OS, we have 3-Schedulers:-

- ① Long-term Scheduler or Job Scheduler.
- ② Short-term " " " CPU "
- ③ Mid-term " " " or Medium-term Scheduler.

→ Long-Term Scheduler :-
It is responsible of creating & bringing new processes into the system.

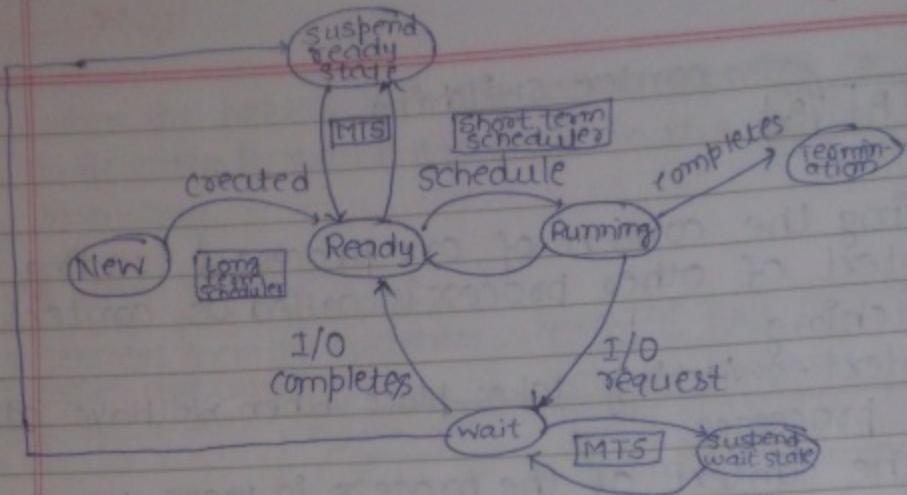
Cosmos

MTS :- mid-term scheduler

classmate

Date _____

Page _____



- Short - Term Scheduler :- is responsible for selecting one of the process from ready state to scheduling onto running state.
- Mid-Term Scheduler :- is responsible for suspending & resuming the processes. The job done by mid-term Scheduler is called as swapping.
- Dispatcher :- is responsible for saving the context of one process & loading the context of another process. Therefore, the context switching is done by the dispatcher.

- ★ If all (or majority) of the processes are I/O bound processes, then running state becomes empty & wait queue increases & hence throughput decreases.
- ★ If all (or majority) of the processes are CPU bound processes ,then each process will take more time in running state, CPU utilization increases, but no. of processes executed per second decreases & hence throughput decreases.

Cosmos

★ Long term Scheduler should select good combination of CPU bound or I/O bound processes in order to get ^{good} throughput of the system.

→ Degree Of multiprogramming :- the no. of jobs present in the mem. at any pt. of time is called as degree of multiprogramming.

The Long-Term Scheduler Controls degree of multiprogramming.

Q. Consider a system with 'N' CPU processors, then what is the min. & max. no. of processes that may be present in ready, running & block states

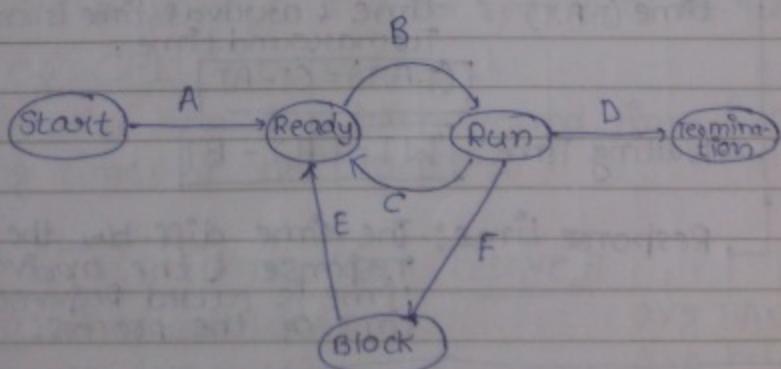
Ans.

Running	Ready	Block
Min. → 0	Min. → 0	Min. → 0
Max. → N	Max. → any no. of processes	Max. → any no. of processes.

(depends upon the size of main mem.)

Min. is 0 because LTS (Long-Term Scheduler) is not creating any process.

Q.



Consider the statements below:-

- If a process makes a transition D, it would result in another process making a transition A immediately.

Cosmos

2. A process P_2 in block state can make a transition E while another process P_1 is in the running state.
3. The OS uses pre-emptive scheduling.
4. The OS uses non-pre-emptive scheduling.
- Which are true?

Ans. (2) & (3)

(1) is not true because there is no relation b/w A & D.

process

→ Arrival time :- The time when the process arrives in the ready state is called arrival time of the process.

→ Completion time :- The time when the process completes its execution is called completion time of the process.

→ Burst time :- The time required for the process for its execution is called burst time of the process.
 no waiting time. \rightarrow (BT = CPU time + I/O time)

, turnaround : The time diff. b/w completion time & arrival time is called Turnaround time.

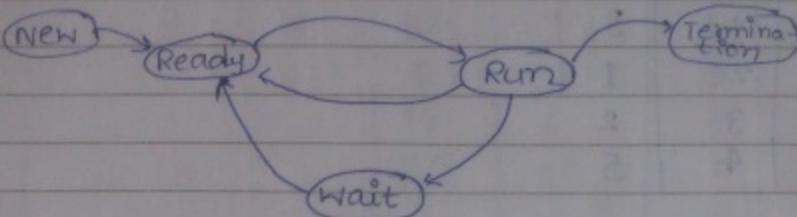
$$(T.A.T) = CT - AT$$

→ Waiting Time :- $WT = T.A.T - BT$

→ Response time :- The time diff. b/w the 1st response & the arrival time is called Response time of the process.

Cosmos

CPU Scheduling



The Short-term Scheduler will use CPU scheduling algorithm in ready state.

- ① When process moves from running to termination (i.e. when process completes)
- ② When process is moving from running to wait state (i.e. when process requires I/O operation.)
- ③ When the process is moving from running to ready (i.e. when time slice expires or the low priority is pre-empted by high priority process.)
- ④ When the process moves from New to ready.
(This takes place because the new process can be a high priority process)
[We have to apply CPU scheduling every time a new process is created.]
- ⑤ When the process moves from wait to ready state (because of similar reason of pt. 4).

Goal of CPU Scheduling:-

To minimize the avg. turnaround time & avg. waiting time of the processes.

① FCFS (First Come First Serve)

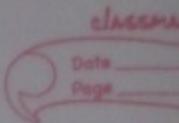
- Criteria: Arrival Time.
- Mode: non-preemptive

$$\text{avg TAT} = ?$$

$$\text{avg WT} = ?$$

★ As criteria is arrival time, when two processes with same arrival time, then schedule that process with lowest process id.

Cosmos



P. No.	AT	BT
1	0	4
2	1	3
3	2	1
4	3	2
5	4	5

Avg. TAT = $\frac{4+6+6+7+11}{5} = \frac{34}{5} = 6.8$
 Gantt Chart

P ₁	P ₂	P ₃	P ₄	P ₅	
0	4	7	8	10	15

$$\text{avg WT} = \frac{0+3+5+5+6}{5} = \frac{19}{5} = 3.8$$

E.T. - A.T.

Q.	P. No.	A.T.	B.T.	C.T.	TAT	W.T. → TAT-BT
1	6	4	22	16	12	
2	2	5	7	5	0	
3	3	3	10	7	4	
4	1	1	2	1	0	
5	4	2	12	8	6	
6	5	6	18	13	7	

	P ₄	P ₂	P ₃	P ₅	P ₆	P ₁	
0	1	2	7	10	12	18	22

$$\text{Avg. TAT} = \frac{50}{6} = 8.33$$

$$\text{Avg. WT} = \frac{29}{6} = 4.833$$

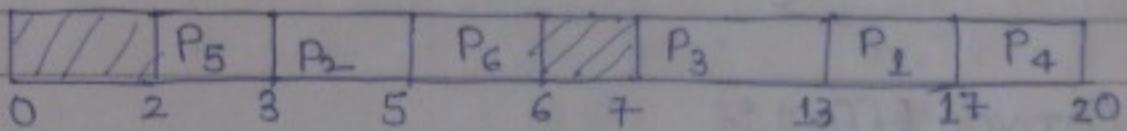
Cosmos

classmate

Date _____

Page _____

Q.	P.No.	A.T.	B.T.	C.T.	T.A.T.	W.T.
1	✓ 8	4	17	9	5	
2	✓ 3	2	5	2	0	
3	✓ 7	6	13	6	0	
4	10	3	20	10	7	
5	✓ 2	1	3	1	0	
6	✓ 3	1	6	3	2	

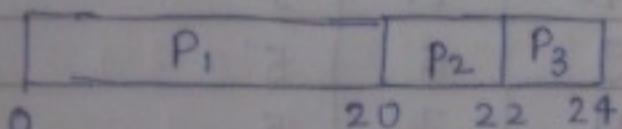


$$\text{Avg. TAT} = \frac{31}{6} = 5.16$$

$$\text{Avg. WT} = \frac{14}{6} = 2.3$$

Note:- If the arrival times of the processes are matching then schedule the process with lowest process id.

Q.	P.No.	A.T.	B.T.	C.T.	T.A.T.	W.T.
1	0	20	20	20	20	0
2	1	2	22	21	19	
3	2	2	24	22	20	



$$\text{Avg. WT.} = \frac{19+20}{3} = 13.$$

Cosmos

classmate

Date _____
Page _____

P.No.	A.T.	B.T.	C.T.	T.A.T.	W.T.
1	0	2	2	2	0
2	1	2	4	3	1
3	2	20	24	22	2

P ₁	P ₂	P ₃
0	2	4

$$\text{Avg. W.T.} = \frac{3}{3} = 1$$

★★ Convoy Effect :-

In FCFS, if the 1st process is having a large burst time, then it will have a huge impact on the avg. waiting time of all the remaining processes. So the effect is called as the Convoy effect.

Shortest Job First

Criteria:- Burst Time

Mode:- Non-preemptive & Pre-emptive both

Non-Preemptive :-

Q.	P.No.	A.T.	B.T	C.T.	T.A.T.	W.T.
	1	1 ✓	7	8	7	0
	2	2 ✓	5	16	14	9
	3	3 ✓	1	9	6	5
	4	4 ✓	2	11	7	5
	5	5 ✓	8	24	19	11

P ₁	P ₂	P ₃	P ₄	P ₅
0	1	8	9	11

Cosmos

classmate

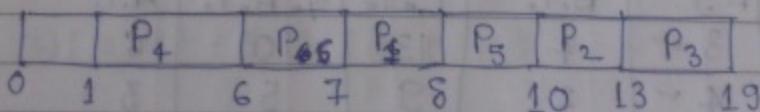
Date _____
Page _____

$$\text{Avg. T.A.T.} = \frac{53}{5} = 10.6$$

$$\text{Avg. W.T.} = \frac{30}{5} = 6$$

Note:- If the burst times of the processes are same
★ then schedule the process with lowest arrival time.

Q.	P.No.	A.T	B.T.	C.T.	T.A.T.	W.T.
	1	6 ✓	1	7	2	1
	2	3 ✓	3	13	10	7
	3	4	6	19	15	9
	4	1	5	6	5	0
	5	2 ✓	2	10	8	6
	6	5 ✓	1	7	2	1



$$\text{Avg. TAT} = \frac{42}{6} = 7$$

$$\text{Avg. WT} = \frac{24}{6} = 4$$

Cosmos

Preemptive

P.No.	A.T.	B.T.	C.T.	T.A.T.	W.T.
1	0	76	19	19	12
2	1	54	13	12	7
3	2 ✓	321	6	4	1
4	3 ✓	1	4	1	0
5	4 ✓	2	9	5	3
6	5 ✓	1	7	2	1

	P ₁	P ₂	P ₃	P ₄	P ₅	P ₃	P ₆	P ₅	P ₂	P ₁
B _u	1	2	3	4	5	6	7	9	13	19

$$\text{Avg. T.A.T} = \frac{43}{6} = 7.16$$

$$\text{Avg. W.T.} = \frac{24}{6} = 4$$

Q.	P.No.	A.T.	B.T.	C.T.	T.A.T.	W.T.
	1	3 ✓	4	13	10	6
	2	4 ✓	2	9	5	3
	3	5 ✓	1	7	2	1
	4	2	6	19	17	11
	5	1	8 7	26	25	17
	6	2 ✓	4 21	6	4	0

	P ₅	P ₆	P ₆	P ₆	P ₃	P ₈	P ₁	P ₄	P ₅
0	1	2	3	4	5	6	7	9	13 19

$$\text{Avg. T.A.T} = \frac{63}{6} = 10.5$$

$$\text{Avg. W.T.} = \frac{38}{6} = 6.33$$

Cosmos

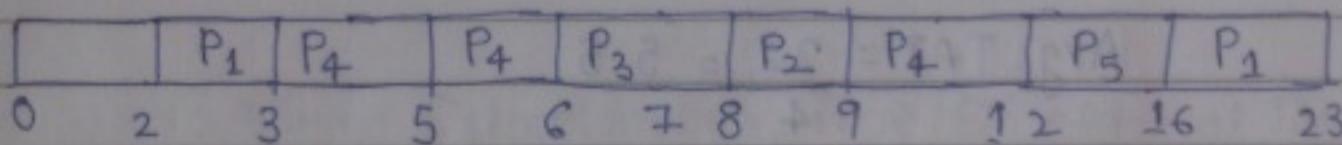
classmate

Date _____

Page _____

★ If we have same burst time, then we will check the burst arrival time, if arrival times are also same, then choose process with lowest process id.

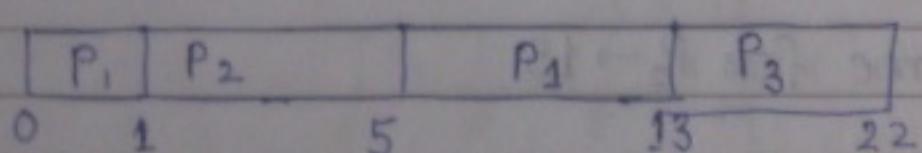
Q.	P.No.	A.T.	B.T.	C.T.	T.A.T.	W.T.
	1	2	8 7	23	21	13
	2	7 ✓	1	9	2	1
	3	6 ✓	2 1	8	2	0
	4	3 ✓	8 4 3	12	9	3
	5	5	4	16	11	7



$$\text{Avg. TAT} = \frac{45}{5} = 9$$

$$\text{Avg. WT} = \frac{24}{5} = 4.8$$

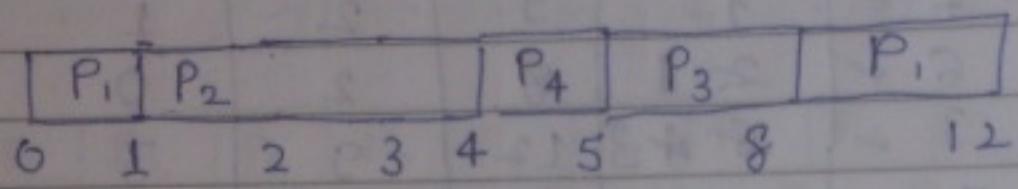
Q.	P.No.	A.T.	B.T.	C.T.	T.A.T.	W.T.
	1	0	9 8	13	13	4
	2	1	4	15	4	0
	3	2	9	22	20	11



$$\text{Avg. WT.} = \frac{15}{3} = 5$$

Cosmos

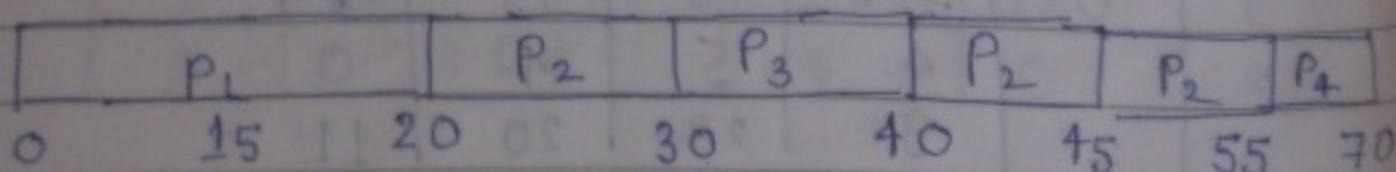
Q.	P.No.	AT	BT	C.T.	T.A.T.	W.T.
1	0	54	12	12	?	0
2	1 ✓	321	4	3	0	3
3	2	3	8	6	3	0
4	4 ✓	1	5	13	01	0



$$\text{Avg. WT.} = \frac{10}{4} = 2.5$$

$$\text{Avg. TAT} = \frac{22}{4} = 5.5$$

Q.	P.No.	AT	BT	C.T.	T.A.T.	W.T.
1	0 ✓	205	20	20	20	0
2	15 ✓	251510	55	40	20	15
3	30 ✓	10	40	10	0	0
4	45	15	70	25	10	0



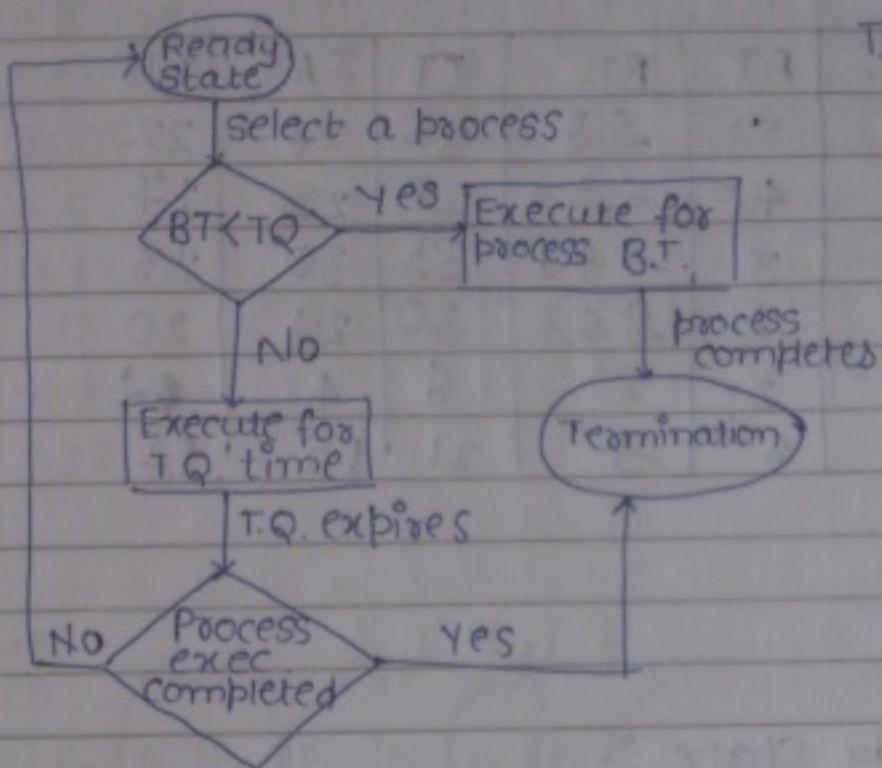
Waiting Time for $P_2 \rightarrow 15$

Cosmos

classmate

Date _____
Page _____

Round-Robin Scheduling:-



TQ → Time Quantum or Time slice

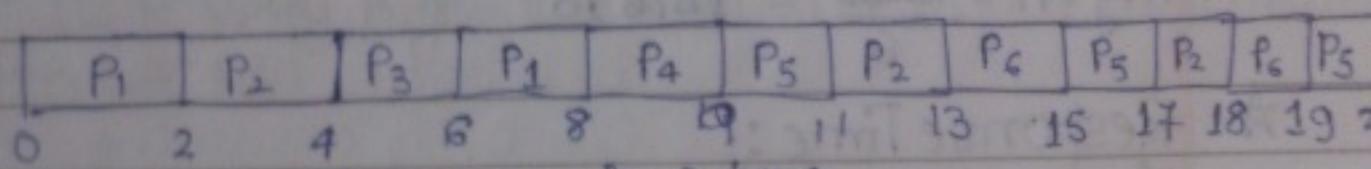
Criteria: Time Quantum Or Time Slice, Arrival Time

Mode: preemptive

* When arrival time of the processes are same, we schedule that process which has lower pid.

Q. T.Q. = 2

P.No.	A.T.	B.T.	C.T.	T.A.T.	W.T.
1	0 ✓	4 2	8	8	6
2	1	5 3 1	18	17	12
3	2	2	6	4	2
4	3 ✓	1	9	6	5
5	4	6 6 4 2	21	17	11
6	6	3 1	19	13	12



Ready State :-

We write P1 here because P1 doesn't gets completed at t=2, we write P1 after P2 & P3

P1, P2, P3, P1, P4, P5, P2, P6, P5, P2, P6, P5
 we wrote this pid if it is the only process at t=0. after t=2 P1 is executed for t=2 & at t=2 P2 & P3 arrive

Q. $TQ = 3$

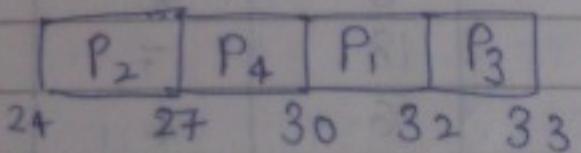
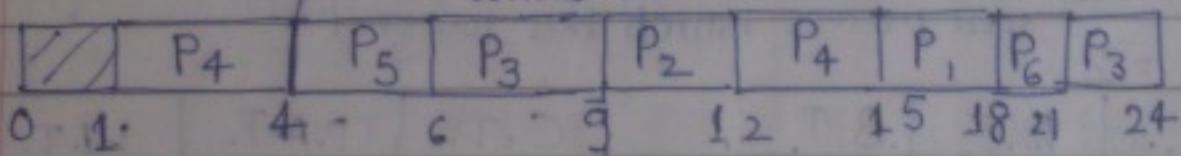
P.No.	AT	BT	CT	TAT	WT
1	5 ✓	5 8	32	27	22
2	4 ✓	6 3	27	23	17
3	3	7 4 1	33	30	23
4	1 ✓	8 6 3	30	29	20
5	2 ✓	2	6	4	4 2
6	6 ✓	3	21	15	12

Cosmos

Ready State :-

$P_4 P_5 P_3 P_2 P_4 P_1 P_6 P_3 P_2 P_4 P_1 P_3$

Context Switching takes place here.



$$\text{Avg TAT} = \frac{128}{6} = 21.3$$

$$\text{Avg WT} = \frac{96}{6} = 16$$

★ Response Time :-

Time diff. b/w 1st response & arrival time.

Response

Cosmos

classmate

Date _____

Page _____

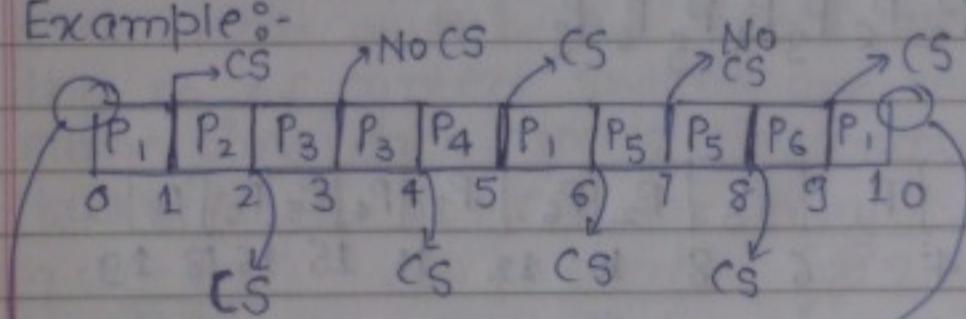
Ready State explanation:-

- ① The process P_4 arrives at $t=1$, & its $BT = 3$ sec, so we put it in ready queue & execute it for 3 sec (i.e. from $t=1$ to $t=4$).
- ② From $t=1$ to $t=4$, 3 new processes arrive in order P_5, P_3, P_2 , so we place P_5, P_3 & P_2 in the ready queue, now process P_4 is not complete so we put it after P_2 , so now ready queue contain P_4, P_5, P_3, P_2, P_4 .
- ③ Now, we will execute P_5 from $t=4$ to $t=6$ (as $BT = 2$ sec) & it gets completed, so we don't put it in ready queue (at $t=6$ P_1 & P_6 also arrives, so we put P_1 in ready queue).
- ④ Now, we execute P_3 from $t=6$ to $t=9$ [but P_3 gets completed], so we put P_3 back in the queue; now ready queue is $P_4, P_5, P_3, P_2, P_4, P_1, P_6, P_3$ & so on.

Process Response Time
(1st response - AT)

$$\begin{aligned}P_1 & 15 - 5 = 10 \\P_2 & 9 - 4 = 5 \\P_3 & 6 - 3 = 3 \\P_4 & 1 - 1 = 0 \\P_5 & 4 - 2 = 2 \\P_6 & 18 - 6 = 12\end{aligned}$$

Example:-



CS:- Context Switching

Not considered for context switching

- * In Round Robin if TQ is less, then no. of context switches will increases & response time decreases. if TQ is more, no. of context switches decreases & response time increases.
- * If TQ is very high, then Round-Robin degenrates into FCFS.

Cosmos

TQ = 2

Q.	P.No.	A.T.	B.T.	CT	TAT	WT	RT
1	3 ✓	2	6	3	1	1	
2	2	42	10	8	4	0	
3	6	32	19	13	11	6	
4	8 ✓	1	15	7	6	6	
5	4 ✓	31	16	12	3	2	
6	5 ✓	42	18	13	9	5	

ready state :-

P₂ P₁ P₅ P₂ P₆ P₃ P₄ P₅ P₆ P₃

	P ₂	P ₁	P ₅	P ₂	P ₆	P ₃	P ₄	P ₅	P ₆	P ₃	
0	2	4	6	8	10	12	14	15	16	18	20

$$\text{Avg. TAT} = \frac{56}{6} = 9.33$$

$$\text{Avg. WT} = \frac{40}{6} = 6.6$$

$$\text{Avg. response time} = \frac{20}{6} = 3.33$$

Consider a system with 4 jobs P₁, P₂, P₃, P₄ arriving in ready queue in same order at t=0. The B.T. respect. are 4, 1, 8, 1. Then what is the C.T. of P₁ assuming round-robin scheduling with TQ=1?

Cosmos

classmate

Date _____
Page _____

$$P_1 \quad 4 \ 3 \ 2 \ 1$$

$$P_2 \quad 1 \ \checkmark$$

$$P_3 \quad 8 \ 7 \ 6 \ 5$$

$$P_4 \quad 1 \ \checkmark$$

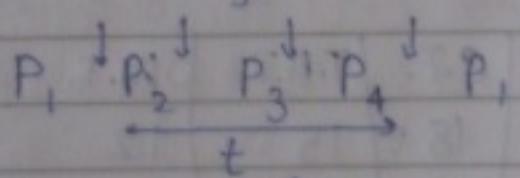
$P_1, P_2, P_3, P_4, P_1, P_3, P_1, P_3$

P_1	P_2	P_3	P_4	P_1	P_3	P_1	P_3	P_1
6	1	2	3	4	5	6	7	8

Q. Consider 'n' processes sharing the CPU in round robin fashion. If the context switching time is 's' then what must be the Time Quantum "q" such that the no. of context switches are reduced, but at the same time each process is guaranteed to get its turn/job at the CPU for every 't' sec. of time

$$(a) q = \frac{t - ns}{n+1} \quad (b) q = \frac{t + ns}{n-1} \quad (c) q = \frac{t - ns}{n-1} \quad (d) q = \frac{t - ns}{n+1}$$

Ans. ★ Assume the arrival time of all processes to be 0 & Burst time to be very large.

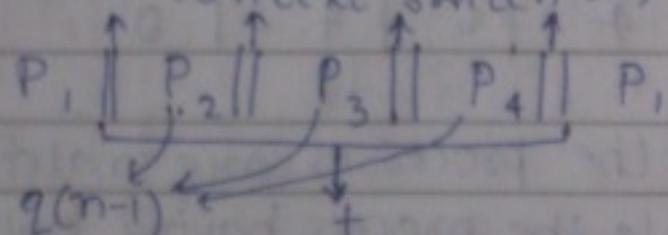


$ns \rightarrow$ time for context switches

$$t = ns + q(n-1)$$

$$\frac{t - ns}{n-1} = q$$

context switch $\rightarrow ns$



$$\therefore t = ns + q(n-1)$$

$[(n-1) \text{ time quantum}]$

Date

Date _____

Page _____

17.06.12

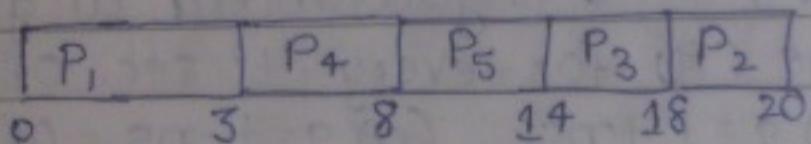
Cosmos

Longest Job First

Criteria:- Burst Time

Mode:- Non-Preemptive & Pre-emptive

Q.	P. No.	A.T.	B.T.	C.T.	T.A.T	W.T.
Non-preemptive)	1	0 ✓	3	3	3	0
	2	1 ✓	2	20	19	17
	3	2 ✓	4	18	16	12
	4	3 ✓	5	8	5	0
	5	4 ✓	6	14	10	4



$$\text{Avg TAT} = \frac{53}{5} = 10.6$$

$$\text{Avg WT} = \frac{33}{5} = 6.6$$

Q.	P. No.	A.T.	B.T.	C.T.	T.A.T.	W.T.
	1	3 ✓	3	18	15	12
	2	1 ✓	1	2	1	0
	3	2 ✓	3	15	13	10
	4	4	2	20	16	14
	5	6 ✓	6	12	6	0
	6	2 ✓	4	6	4	0

Note:- The B.T. of the processes are matching, then schedule the process having lowest arrival time.

Cosmos

classmate

Date _____
Page _____

	P ₂	P ₆	P ₅	P ₃	P ₁	P ₄
0	1	2	6	12	15	18

$$\text{Avg. TAT} = \frac{55}{6} = 9.16 \approx 9.1$$

$$\text{Avg. WT} = \frac{36}{6} = 6$$

Pre-emptive

②	P.N.O.	AT	BT	CT	T.A.T.	W.P.	C.T.	T.A.T.	W.T.
	1	1	21	21	20	18	18	17	15
	2	2	4 3 2	20	18	14	19	17	13
	3	3 ✓	8 8 4	17	14	8	20	17	11
	4	4 ✓	8 8 4 2	12	8	0	21	17	9

	P ₁	P ₂	P ₃	P ₄	P ₃	P ₂	P ₁
0	1	2	3	4	12	17	20

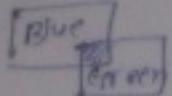
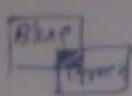
Answe^r: Avg. TAT = $\frac{60}{4} = 15$

$$\text{Avg. WT} = \frac{40}{4} = 10$$

	P ₁	P ₂	P ₃	P ₄	P ₃	P ₄	P ₃	P ₄	P ₂	P ₃	P ₄	P ₂	P ₃	P ₄	
0	1	2	3	4	7	8	9	10	11	12	13	14	15	16	
	P ₁	P ₂	P ₃	P ₄											
	17	18	19	20	21										

Correct Answer:- Avg. TAT = 17 Avg. WT = $\frac{48}{4} = 12$

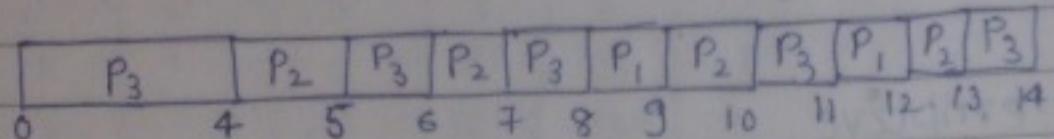
Cosmos



classmate

Date _____
Page _____

Q.	P.No.	AT	BT	C.T.	T.A.T.	W.T.
1	0	2 1	12	12	12	10
2	0	2 3 2 1	13	13	13	9
3	0	8 4 2 3 2 1	14	14	14	6



$$\text{Avg TAT} = 13$$

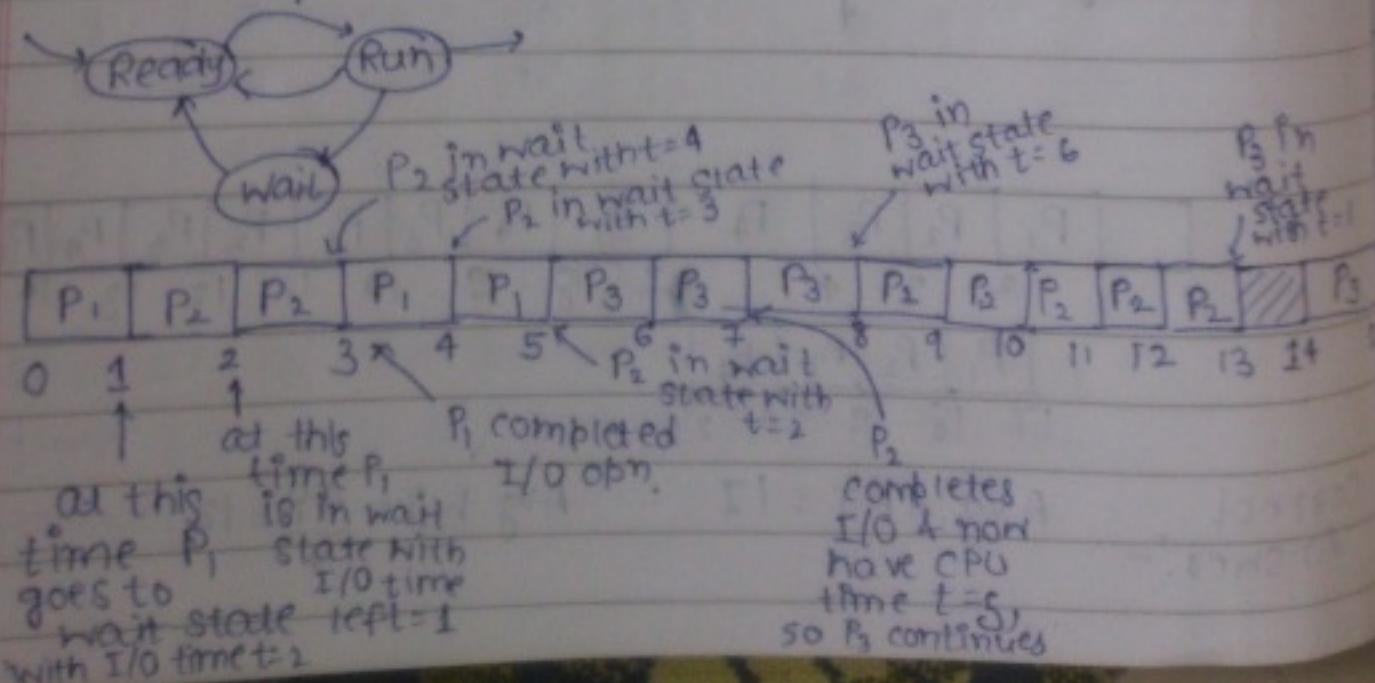
$$\text{Avg. WT} = \frac{25}{3} = 8.33$$

Preemptive Shortest Job First :-

Q	P.No.	AT	Execution Time	CPU Time	I/O Time	CPU Time	C.T.	T.A.T.
1	0	1	10	10	0	0	5	5
2	1	2	10	4	0	5	13	12
3	0	3 2 1	11	6	5	8	22	20

Note:- ① Process first spends CPU time followed by I/O time & followed by CPU time again.

② I/O of the processes can be overlapped as much as possible.



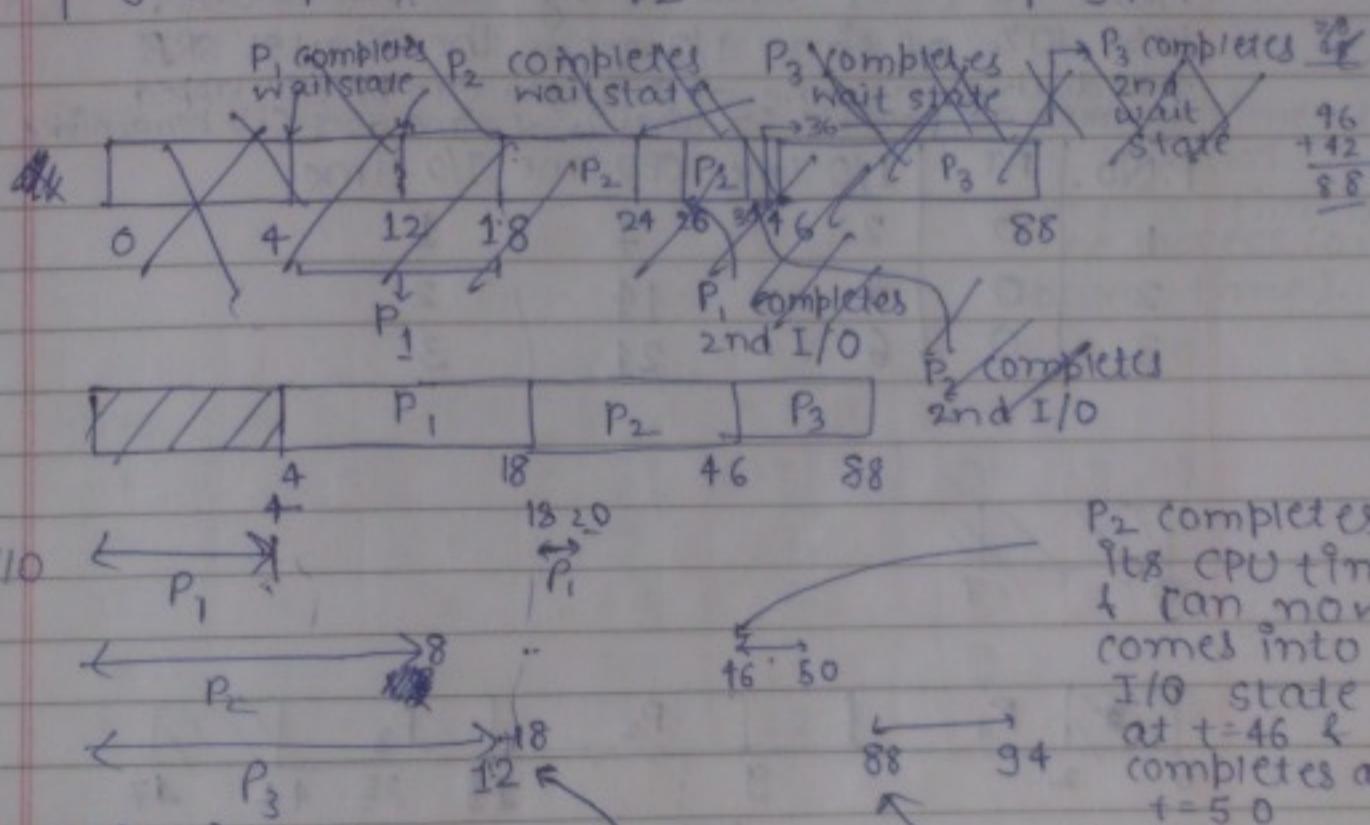
Cosmos

classmate

Date _____
Page _____

* We don't calculate T.A.T. & W.T. in these questions.

Q.	P.No.	AT	I/O time	CPU time	I/O time	C.T.
	1	0	40	40	20	20
	2	0	80	280	4	50
	3	0	120	420	6	94



At this time
all the I/O's
can be overlapped
& hence P₁ I/O time,
P₂ I/O time, P₃ I/O time
overlaps.

P₁ completes
CPU time at t=18
& its I/O time
can start at t=18
& completes at t=20

P₃ Completes
its CPU
time at t= 88 &
is ready for
its I/O op
at t=88 &
completes at
t=94.

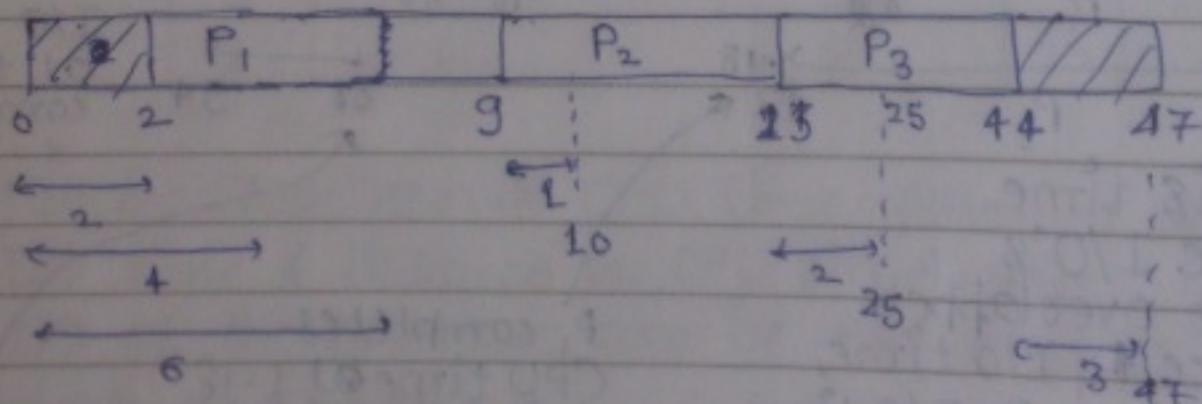
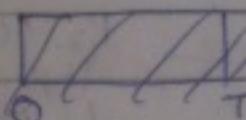
Cosmos

classmate

Date _____
Page _____

- Q. Consider 3 processes all arriving at time t=0 with total execution time of 10, 20, 30 time units resp. Each process spends the 1st 20% of execution time doing I/O, the next 70% of time doing computation & last 10% of time I/O again, the OS uses SJPF scheduling, assume all I/O, computation can be overlapped as much as possible, for what % of time does CPU remain idle.

P.No.	AT	I/O time	CPU time	I/O time
1	0	2	7	1
2	0	4	14	2
3	0	6	21	3



$$\text{CPU-% idle} = \frac{5}{47} \times 100 = \frac{500}{47} \approx 10.6\%$$

$$\frac{\frac{10.6}{47}}{500} \times 100 = \frac{10.6}{300}$$

Cosmos

classmate

Date _____

Page _____

Highest Response Ratio Next (HRRN)

It favours the shorter jobs & also limits the waiting time of longer jobs

Criteria:- Response Ratio

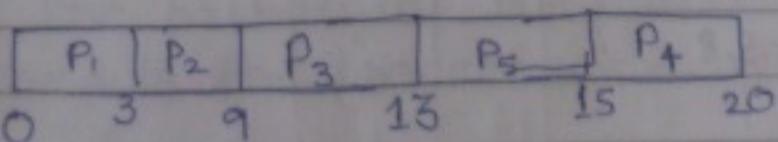
Mode:- Non-preemptive

$$\text{Response Ratio} = \frac{WT + S}{S}$$

where WT:- Waiting time of the process

S:- service time of the process (or Burst time).

①	P.No.	A.T.	B.T.	C.T.	T.A.T.	W.T.
	1	0	3	3	3	0
	2	2	6	9	7	1
	3	4	4	13	9	5
	4	6	5	20	14	9
	5	8	2	15	17	5



$$RR(P_3) = \frac{(9-4)+4}{4} = \frac{9}{4} \quad \text{WT} = 9 - AT$$

$$RR(P_4) = \frac{(9-6)+5}{5} = \frac{8}{5}$$

$$RR(P_5) = \frac{(9-8)+2}{2} = \frac{3}{2}$$

$$\text{Avg. TAT} = \frac{40}{5} = 8$$

$$\text{Avg. WT} = \frac{20}{5} = 4$$

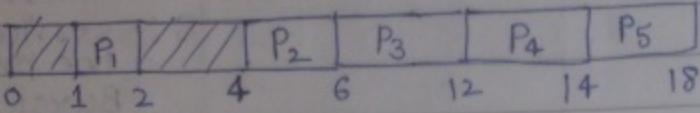
$$RR(P_4) = \frac{(13-6)+5}{5} = \frac{12}{5} = 2.4$$

$$RR(P_5) = \frac{(13-8)+2}{2} = \frac{7}{2} = 3.5$$

Cosmos

(2)

P.No.	AT	BT	CT	TAT	WT
1	1 ✓	1	2	1	0
2	4 -	2	6	2	0
3	5 ✓	6	12	7	1
4	6	2	14	8	6
5	7	4	18	11	7

at
t=6

$$RR(P_3) = \frac{(6-5)+6}{6} = \frac{7}{6} \checkmark$$

$$RR(P_4) = \frac{(6-6)+2}{2} = 1$$

at t=12

$$RR(P_4) = \frac{(12-6)+2}{2} = \frac{8}{2} = 4$$

$$RR(P_5) = \frac{(12-7)+4}{4} = \frac{9}{4}$$

$$Avg TAT = \frac{29}{5} = 5.8$$

$$Avg WT = \frac{14}{5} = 2.8$$

Cosmos

classmate

Date _____

Page _____

Priority Based Scheduling

Criteria:- Priority

Mode :- Non-preemptive

Priority	P.No.	A.T.	B.T.	C.T.	TAT	WT.	Assumption → Highest no. → Highest priority
4	1 ✓	0	4	4	4	0	
5	2 ✓	1	5	10	15	10	
7	3 ✓	2	1	5	3	2	
2	4 ✓	3	2	18	15	13	
1	5	4	3	21	17	14	
6	6 ✓	5	6	11	16	0	

P ₁	P ₃	P ₆	P ₂	P ₅	P ₄	P ₅
0	4	5	10	11	16	21

$$\text{Avg. TAT} = \frac{60}{6} = 10 \quad \text{Avg. WT} = \frac{30}{6} = 5$$

Note:- If the priority of the process are matching, then schedule the process with lowest arrival time.

P.No.	AT	BT	Priority	CT	TAT	WT	Avg TAT = $\frac{80}{6} = 7$
1	4	6	4	10	13	3	
2	6 ✓	3	5	11	45	8	
3	3 ✓	4	6	8	4	0	Avg WT = $\frac{21+3}{6} = 4$
4	2 ✓	2	6	4	2	0	
5	1 ✓	1	7	2	1	0	
6	2 ✓	2	3	18	17	15	

P ₅	P ₄	P ₃	P ₂	P ₁	P ₆
0	1	2	4	8	16

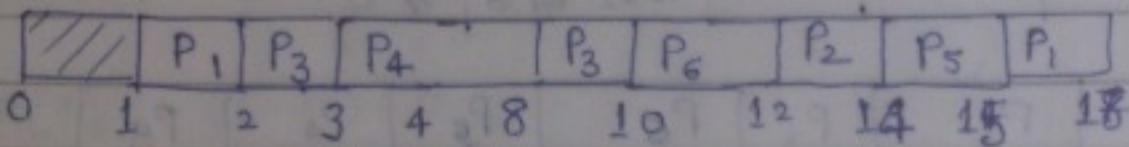
Cosmos

classmate

Date _____
Page _____

Q Now, consider round robin scheduling
Preemptive

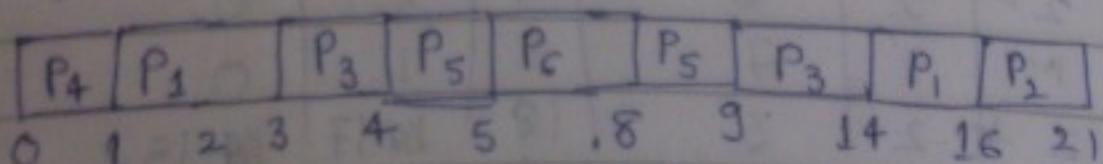
P.NO.	AT	BT	Priority	CT	TAT	WT
1 ✓	1	4	4	18	17	13
2 ✓	2	2	5	14	12	10
3 ✓	2	3	7	10	8	5
4 ✓	3	5	8	8	5	0
5 ✓	3	1	5	15	12	11
6 ✓	4	2	6	12	8	6



$$\text{Avg TAT} = 162/6 = 10.3$$

$$\text{Avg WT} = 45/6 = 7.5$$

Q.	P.NO.	AT	BT	Priority	CT	TAT	WT
	1	1	4	5	16	15	11
	2	2	5	2	21	19	14
	3	3	8	6	14	11	5
	4 ✓	0	4	4	11	11	0
	5 ✓	4	2	7	9	5	3
	6 ✓	6	3	8	8	3	0



$$\text{Avg TAT} = \frac{54}{6} = 9$$

$$\text{Avg WT} = \frac{33}{6} = 5.5$$

Cosmos

classmate

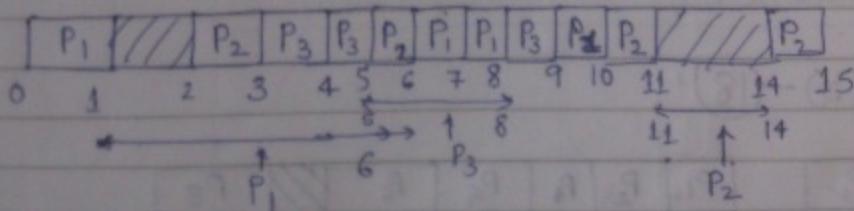
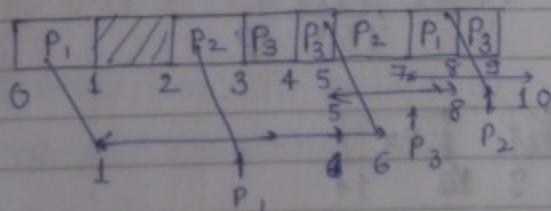
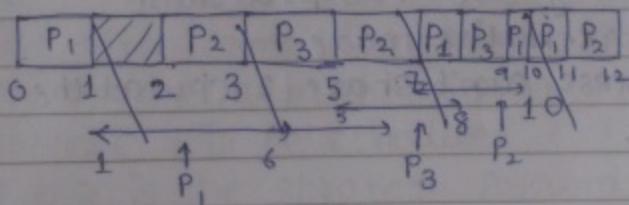
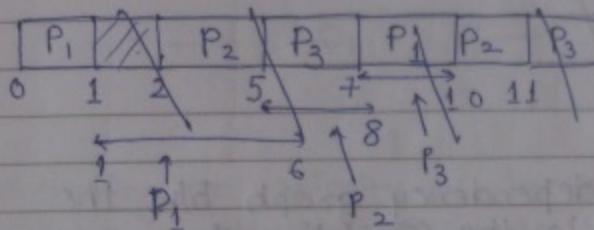
Date _____

Page _____

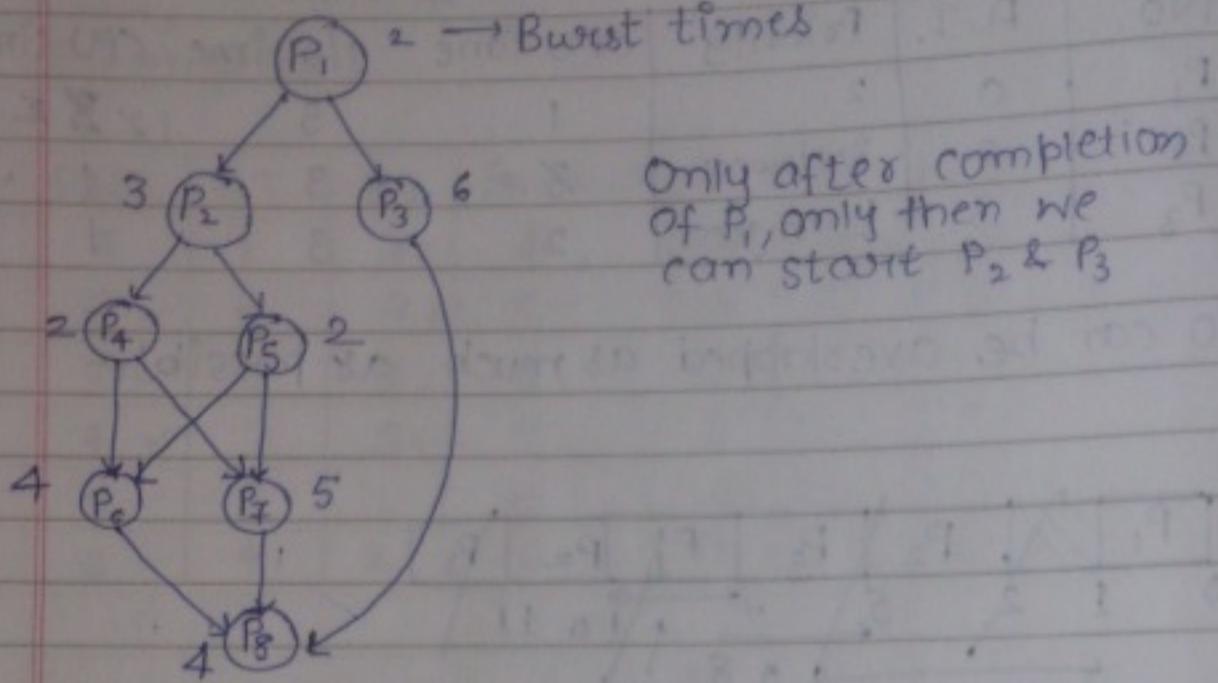
Burst time

Q.	P. No.	A.T.	Priority	CPU time	I/O time	CPU time	CT
	P ₁	0	2	1	5	2	10
	P ₂	2	3 → lowest	3	2	1	10
	P ₃	3	1 → highest	2	3	1	9

I/O can be overlapped as much as possible



Cosmos



Q. Consider the dependency graph b/w the processes what is the completion time of all the processes using two - CPU processors?

Note:- (1) USE non-preemptive mode

(2) One process can't share 2-CPU at the same time.

Ans.

CPU 1 :-	P_1	P_2	P_4	P_5	P_6	P_8
	0	2	5	7	9	13 14 18

CPU 2 :-	P_3	P_7	P_8
	0	2	8 9 12 14

Processors CT

P_4

Ans. \rightarrow (b) - 18

P_1	P_1	P_2	P_4	P_5	P_6	P_8
0	2	0	2	5	7 8 9	13 14 18

Cosmos

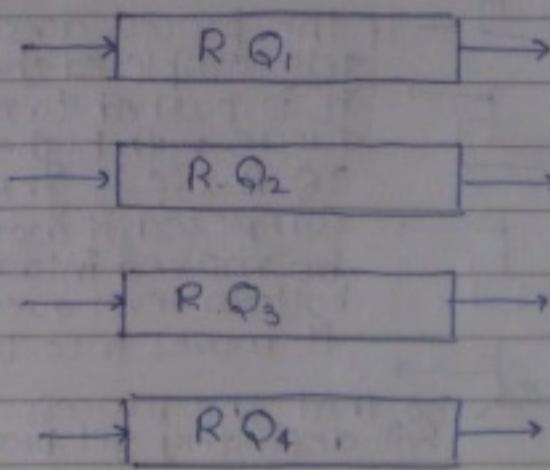
classmate

Date _____

Page _____

Multilevel Queue Scheduling

(Distributing processes into multiple ready queues.)



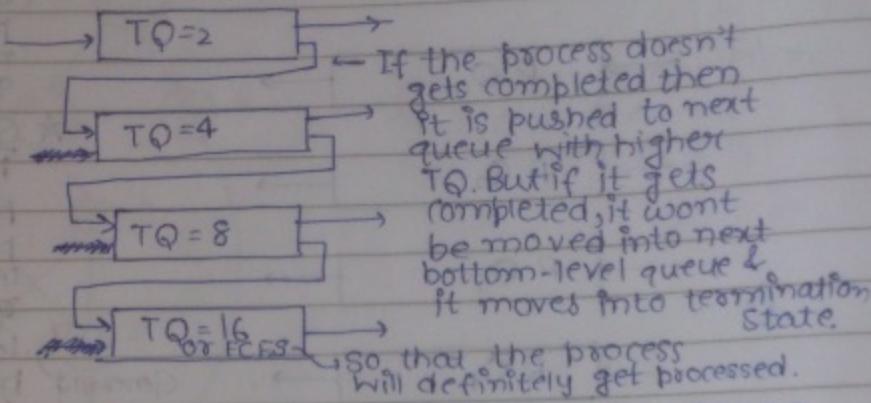
* Generally high priority processes are placed in top level ready queue

* Generally low priority processes are placed in bottom-level ready queue.

- * Depending on priority of process, in which particular ready queue the process has to be placed will be decided.
- * Only after completion of all the processes in top-level ready queue, the further level ready queue processes will be scheduled.
- * If this is the strategy followed, then the processes which are placed in bottom-level ready queue will suffer from starvation.
- * Starvation:- The indefinite waiting of a process is known as starvation.

Multilevel Feedback Queue Scheduling

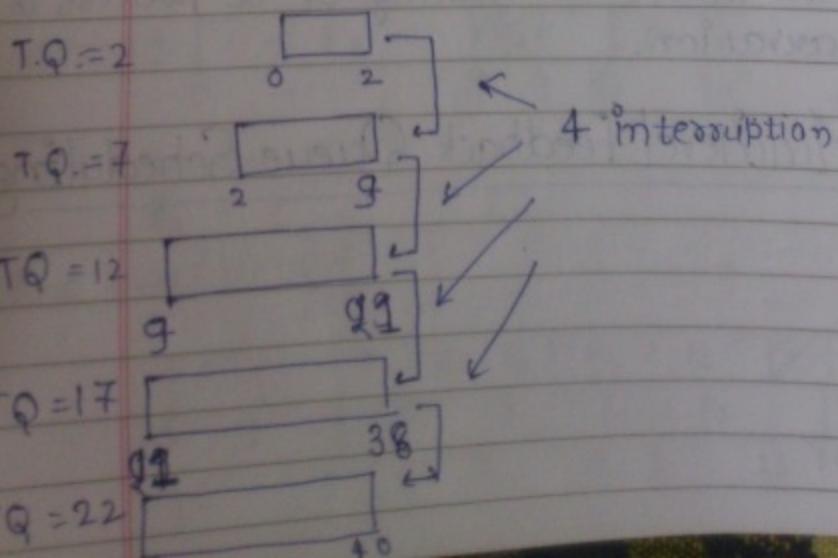
Cosmos



It avoids the starvation & at the same time gives pref. to high priority process.

- Q. Consider a system with having process with BT = 40 s. If the Queue T.Q.=2 & at each level the T.Q. increases by 5 time units. How many times the job will be interrupted & from which queue it will terminate?

- (a) 5, 4
- (b) 4, 3
- (c) 4, 5
- (d) 3, 4



Cosmos

classmate

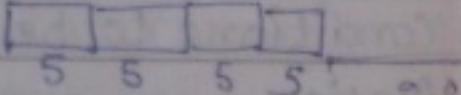
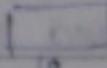
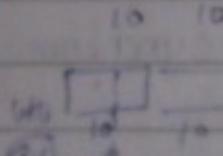
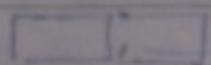
Date _____

Page _____

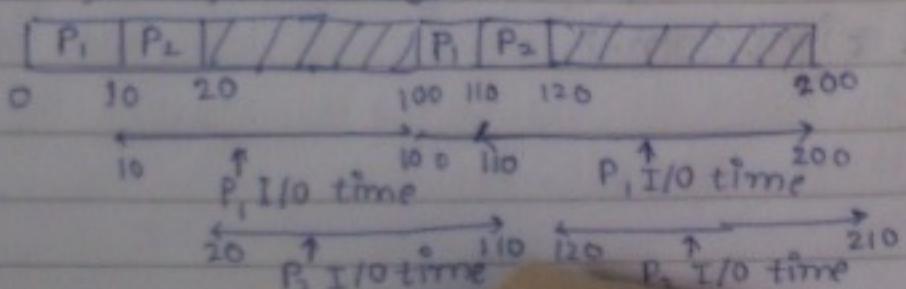
Algorithm	Starvation
1. FCFS	No
2. NP \rightarrow SJF	Yes
3. SRTF (Shortest remaining time first)	Yes
4. NP \rightarrow LTF	Yes
5. LRTF [P \rightarrow LTF]	Yes
6. Round Robin	No
7. NP priority	Yes
8. Pre-priority	Yes
9. HRRN	No
10. M.L.Q.	Yes
11. M.L.F.Q.	No

Q. Consider a system having only 2 processes, both of which alternate 10 sec of CPU burst/time with 90 sec. of I/O time. Both the processes were created nearly at the same time. The I/O of both the processes can proceed in parallel. Which of the following scheduling strategy will result in least CPU utilization over a long period of time

- (a) FCFS
- (b) SRTF
- (c) Static priority scheduling with diff. priorities of the process
- (d) Round Robin with time quantum of 5 sec.

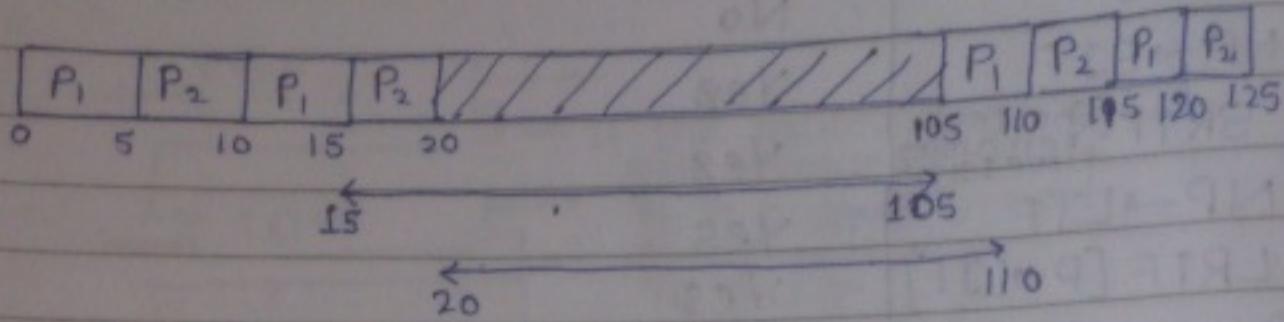


FCFS, SJF, Priority :-



Cosmos

RR :-



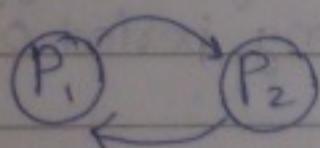
W.T.(for (a),(b),(c)) \rightarrow 80

W.T. (for (d)) → 85

Synchronisation

Co-operative process :-

The execution of 1 process affected effects the other process or effected by the other process, then those 2 processes are said to be co-operative process, otherwise they are independent process.



This happens when they have shared variables, shared resources, etc.

1. Problems which arises not having Synchronisation b/w the processes.
 2. Conditions to be followed to achieve synchronisation.
 3. Solutions off for synchronisation. (wrong soln & right soln.)

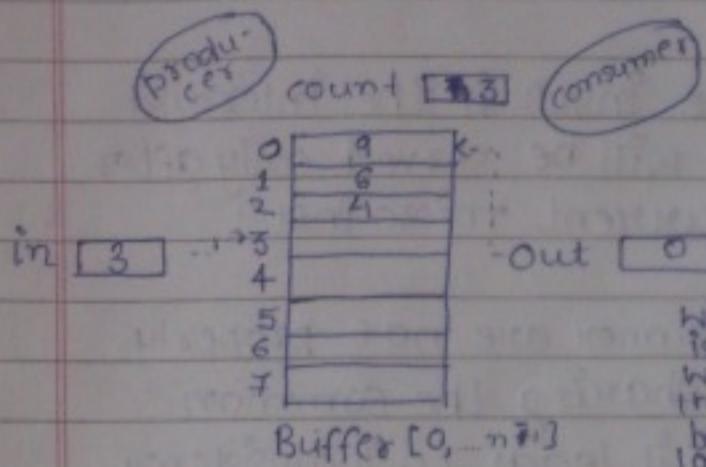
Cosmos

classmate

Date _____

Page _____

Producer - Consumer Problem :-



void consumer(void)

```
{ int ItemC;
  while(true)
```

```
{ while(count == 0);
  itemC = Buffer[out];
  out = (out + 1) Mod N;
  count = count - 1;
```

process_item(itemC);

}

}

```
int count = 0;
void producer(void)
{ int ItemP;
  while(true)
  { produce_item(ItemP);
    while(count == N);
    if(Buffer[in] == ItemP);
    in = (in + 1) Mod N;
    count = count + 1;
  }
}
```

When buffer is full, i.e. when count = N, then this loop + producer next 3 lines are not executed, so producer can't produce any item till count remains N.

I. load $R_p \leftarrow M[count]$
II. INC R_p
III. $M[count] \leftarrow R_p$

when buffer is empty, then next lines are not executed, & hence consumer won't be able to consume anything.

I. $R_c \leftarrow M[count]$
II. DEC R_c
III. $M[count] \leftarrow R_c$

- ★ 'in' is a variable used by the producer in order to identify the next empty slot in the buffer.
- ★ 'out' is a variable used by the consumer to identify from where it has to consume the item.
- ★ 'count' is a variable used by the producer & consumer to identify the no. of filled slots in the buffer at any pt. of time.

Cosmos

Shared Resources:-

1. Buffer
2. Count

- ★ While executing the $inst^n$ if the interrupt comes, the interrupt will be served only after completion of the current micro $inst^n$.
- ★ The producer & consumer are not properly synchronised while sharing the common variable count, hence it leads to inconsistency.

Printer Spooler Daemon

- Rⁱ: - resp. process reg.
(every process will have its own registers.)
1. Enter-File :-
 2. load Rⁱ, M[in]
 3. Store SD[Rⁱ], "F-N"
 4. INCR Rⁱ
 5. Store M[in], Rⁱ
- loading the Rⁱ with the memory address of next empty slot in the spooler directory.

in [3]	0	x
	1	y
	2	z
	5	

out

Printer

Spooler-Directory [SD]

[This Spooler - directory contains all the documents to be printed.]

- ★ 'in' is a variable used by all the processes in order to identify the next empty slot in the spooler directory.

Cosmos

classmate

Date _____

Page _____

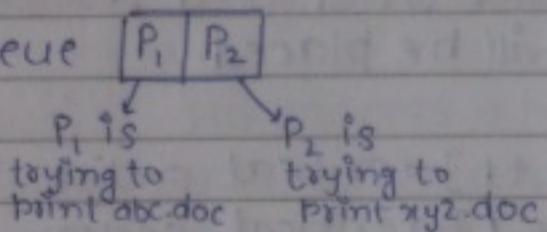
Shared Resources :-

1. In variable
2. Spooler-Directory

* Out is used by printer only to identify from which slot it has to print the document.

Analysis :-

Ready Queue



→ Assume P_1 is executing 1st instn.

$$R_1 [3] \quad M[im] = 3$$

→ Now P_1 is executing 2nd instn.

$$SD[R_1] = SD[3]$$

$$SD[3] \leftarrow abc.doc$$

→ Now P_1 is executing 3rd instn.

$$R_1 \leftarrow R_1 + 1$$

$$R_1 [4]$$

→ Now, assume P_1 is preempted by P_2 at this stage

→ Now P_2 executes 1st instn.

$$R_2 \leftarrow M[im] \quad M[im] = 3$$

$$R_2 \leftarrow 3 \quad R_2 [3]$$

→ Now P_2 executes 2nd instn

$$SD[R_2] = SD[3]$$

$$SD[R_2] \leftarrow SD[3] \leftarrow xyz.doc \quad [abc.doc is overwritten by xyz.doc]$$

∴ So, loss of data occurs.

The processes are not properly synchronised while sharing common variable im, hence loss of data occurs.

Cosmos

★ It is also possible that Deadlock will occur if processes are not properly synchronised.

Definitions:-

1. Critical Section (CS):-

The portion of program text where the shared variables or shared resources or shared data will be placed.

e.g.

$count = count + 1$] → critical section

$if count = count - 1$] → critical section

2. Non-Critical Section:-

The portion of program text where the independent code of the process will be placed.

e.g.

$in = (in + 1) \text{ Mod } N$;] → non-critical section

3. Race Condition:-

The final O/p of any variable depends on execution sequence of ~~on~~ processes, so this is called as race-condition.

Cond'n to be followed to achieve Synchronisation:-

① Mutual Exclusion:-

→ No 2 processes may be simultaneously present inside the critical section at any pt. of time.

Cosmos

classmate

Date _____
Page _____

→ Only 1 process is allowed in critical section at any pt. of time.

② Progress :- No process running outside the critical section should block the other interested process from entering into critical section when critical section is free.

So, any process running in critical section will block other process ~~or~~ trying to enter the critical section.

③ Bounded Wait :- No process should have to wait forever from entering into critical section.

There should be a bound on getting chance for entering into the critical section.

If bounded waiting is not satisfied, there is a possibility of starvation. [if there is no bound, then it can lead to starvation]

④ No assumptions related to h/w or the processor's speed. [soln. is irrespective of h/w or processor's speed]

Solutions :-

① Software type:-

(a) lock variables [No mutual exclusion, no progress]

(b) Strict alteration or Dekker's algo [Mutual exclusion, no progress]

(c) Peterson soln. [Mutual exclusion, progress, bounded waiting]

② Hardware type:-

(a) TSL (Test & Set Lock) instn set

③ OS type :-

(a) counting semaphore

(b) binary semaphore

④ Compiler & Programming Support Type

(a) Monitors

Cosmos

Q10
90%

Date _____
Page _____

Lock Variables

Entry-Section

1. Load $R_i, M[\text{lock}]$
2. CMP $R_i, \#0$
3. JNZ to Step 1
4. Store $M[\text{lock}], \#1$
5. [CS]
6. Store $M[\text{lock}], \#0$

Lock

0

1

CS is
free.

CS is
busy.

$R_i \rightarrow$ resp. process register
(every process will have its own register).

Exit-Section
or remainder section.

Analysis :-

Ready Queue :- $[P_1 | P_2]$

→ Assume lock = 0

→ P_1 executing 1st instn

$$R_1 \leftarrow M[\text{lock}]$$

$$\therefore R_1 = 0 \quad R_1 [0]$$

→ P_1 executing 2nd instn

$$\text{CMP } R_1, 0$$

$$ZF = 1$$

→ P_1 executing 3rd instn

Since $ZF = 1$, \therefore Program Counter = 4

→ At this stage, P_2 preempts P_1

→ Now P_2 executes 1st instn

$$R_2 \leftarrow M[\text{lock}]$$

$$R_2 \leftarrow 0 \quad R_2 [0]$$

→ Now P_2 executes 2nd instn

$$\text{CMP } R_2, 0$$

$$\therefore ZF = 1$$

→ Now, P_2 executes 3rd instn

which makes PC = 4

→ Now P_2 executes 4th instn

Cosmos

classmate

Date _____

Page _____

$M[\text{lock}] \leftarrow 1$
[1] lock

- Now P_2 is executing 5th instⁿ [Critical Section]
→ Now if P_1 comes back to execution starting from instⁿ 4

$M[\text{lock}] \leftarrow 1$
[1] lock

- Now P_1 executes 5th instⁿ & hence enters critical section. (So, both P_1 & P_2 are in critical section).

★ We have proved that both P_1 & P_2 are simultaneous present in the critical section at same time & hence Mutual exclusion is not satisfied, so soln. is bound to be incorrect.

- If the other process preempts just after last instⁿ of critical section & before 6th instⁿ, then no other process wont be able to enter the critical section
- when some process pre-empts a process which is executing its last instⁿ of CS, i.e. just after CS & just before 6th instⁿ, then it wont be able to enter the

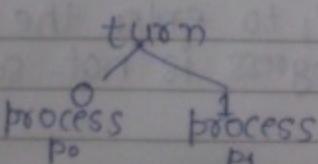
Strict alteration Or Dacker's Algo :-

process p_0 code:-

```
while(true)
    { Non-CS();
        while(turn!=0);
        [CS]
        turn=1;
    }
```

process p_1 code:-

```
while(true)
    { Non-CS();
        while(turn!=1);
        [CS]
        turn=0;
    }
```



Cosmos

Analysis :-

- Assume turn = 0
- Now P₀ is trying to enter Critical Section
- Now turn != 0 is false,
so P₀ enters into critical section.
- Now P₀ is in critical section,
if P₁ is trying to enter critical section
then turn != 1 is true,
so P₁ is not able to enter critical
Section.
- When P₀ completes critical section,
& makes turn = 1.
- turn = 1 makes
turn != 1 false
& P₁ is able to enter critical section
& P₀ is not able to enter critical
section.
So, mutual exclusion is satisfied.
- Now, P₁ is leaving the critical section
& makes turn = 0
& now after sometime P₁ is trying to
enter critical section (critical section
is free).
- turn = 0
& hence, turn != 1 is true
& P₁ is not able to enter the critical
section
but P₀ is not in the critical section
& is hindering P₁ to enter the critical
section, hence progress is not satisfied.

Cosmos

classmate

Date _____

Page _____

★ 2nd Defⁿ of Progress :-

which process will go next into critical section
is decided by those processes that want to go
into critical section.

In this decision, the process in remainder section
the process which is not interested to go into
critical section should not take participation.

Cosmos

(Similar to Strict Alteration)

Q.

Method used by P_1
 $\text{while } (S_1 == S_2);$

[CS]

$S_1 = S_2;$

Method used by P_2
 $\text{while } (S_1 != S_2);$

[CS]

$S_2 = (\text{not}) S_1;$

S_1 & S_2 are two shared (boolean) variables randomly assigned. Consider the methods used by process P_1 & P_2 , which of the below properties are satisfied:

- (a) Mutual Exclusion but not progress.
 - (b) Progress but not mutual exclusion.
 - (c) both M.E. & progress.
 - (d) neither M.E. nor progress.
- * S_1 & S_2 can only have values 0 & 1.

Peterson's Solution:-

```
#define N 2
#define TRUE 1
#define FALSE 0
int turn;
int interested[N];
```

void enter_region(int process)
 { int other;

2. other = ! process;

3. interested[process] = true;

4. turn = process;

5. while (turn == process && interested[other] == true);

the turn is of the
 'process' & not of 'other'
 but 'other' process is
 interested to enter
 section

Q. Why $\text{turn} = \text{process}$?
Ans. Suppose the circumstances are initial, so $\text{interested}[0] \& [1] = \text{FALSE}$
& P enters into entry region & set $\text{interested}[0] = \text{TRUE}$ & $\text{turn} = 0$, let suppose P,
preempts P₀ just before statement 5, i.e. it sets P's $\text{interested}[1] = \text{TRUE}$ &
executes 5th instⁿ, so $\text{turn} = 1$ is true & $\text{interested}[0] = \text{true}$ is also
true, P₀ won't be able to enter CS, now for P₀, $\text{turn} = 0$ is false & hence
true, P₀ wont be able to enter CS & when it leaves CS & execute $\text{interested}[0]$ =
~~TRUE~~, only then P₀ enters into CS, now let suppose P enters

[CS]

`void Leave_region (int process)`

$\{\text{interested}[\text{process}] = \text{FALSE};$

$\}$

into CS & P₀ is not interested, so
 $\text{interested}[0] = \text{FALSE}$, so P₀ enters
into entry region again & set
 $\text{turn} = 1$ & $\text{interested}[1] = \text{TRUE}$
& executes 5th instⁿ, now $\text{turn} =$
 $= 1$ is true but $\text{interested}[0] =$
 $= \text{TRUE}$ is false, so P₀ again enters
into CS & hence satisfies
progress.

- * solution works only for 2 processes.
- * the initial values of $\text{interested}[0]$ & $\text{interested}[1]$ equals to FALSE.

Analysis :-

Cosmos

task

- ① First P₀ will be in critical section (assumption)
 $\therefore \text{process} = 0$
 $\therefore \text{other} = 1$
- ② In 3rd step $\text{interested}[\text{process}] = \text{interested}[0] = \text{true}$
- ③ In 4th step $\text{turn} = \text{process} = 0$.
- ④ In 5th step
 $\text{while} (\text{turn} == \text{process} \& \& \text{interested}[\text{other}] == \text{true})$
 true false
- ⑤ So, P₀ enters into critical section.
- ⑥ Now, let P₁ enter into CS be in critical section, we will check whether P₁ is allowed.
- ⑦ Now for P₁, we call enter-region.
- ⑧ For P₁, $\text{process} = 1$ & $\text{other} = 0$.
- ⑨ In 3rd step, $\text{interested}[1] = \text{true}$.
- ⑩ In 4th step, $\text{turn} = 1$.
- ⑪ In 5th step
 $\text{while} (\text{turn} == \text{process} \& \& \text{interested}[\text{other}] == \text{true})$
 true true
- ⑫ So, P₁ wont be able to enter critical section.
So, Mutual Exclusion is satisfied [When P₁ 1st enters
critical section & then P₀ trying to enter. So M.E.
is satisfied only after checking both cases.]

Cosmos

CLASSMATE

Date _____
Page _____

Checking for progress

- (13) initial values for interested[0] & interested[1] equals false.
- (14) For P₀, s₀
Progress is satisfied.
- (15) If interested[process] = FALSE is not executed,
& process other process preempts, we assume
that "process" is still in the critical section,
& the "other" process can't enter the critical
section. So progress is satisfied.
- ★ If the no. of processes are countable or bounded,
then bounded waiting is satisfied.

Hardware type Of Solutions

- TSL instruction set :-
- ↳ test & set Lock
- This is similar to lock variable
but here we load & change the value of flag in same instrn.
- ✓ TSL Registers flag :- Copies the current value of flag into the register & stores value of 1 into the flag in the single atomic cycle without any pre-emption.

- * The soln. can be used for any no. of processes.

R_i :- Separate Entry-Section

for diff. processes

1. TSL R_i, M[flag]
2. CMP R_i, #0
3. JNZ to Step ①
4. Store M[flag], #0

Analysis :-

① Assuming CS is free & flag is 0.

② Assuming 2 processes in ready queue. (P₁ & P₂) .

③ Let P₁ trying to execute.

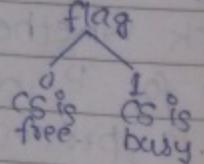
④ P₁ executes 1st Instn.

P₁ ← 0

↳ flag is changed to 1 in same machine cycle.

This step takes place in 1 machine cycle.

These 3 instn. are atomic, which means no process can preempt the execution in b/w.



R_i ← 1
R_i ← 1

flag

R_i ← 0

R_i ← 1

flag

R_i ← 0

- ⑤ Now, P_1 executes 2nd instn, so $R_1 = 0$, $\therefore ZF = 1$,
 so P_1 goes to ~~the~~ critical section, now we will check whether P_2
 comes onto CS while P_1 is still in the CS.
- ⑥ P_2 executes 1st instn.
 $R_2 \leftarrow 1$ & flag becomes 01
- ⑦ P_2 executes 2nd instn
 $ZF = 0$, When it executes 3rd instn, it will go back to 1st instn
 [Now, when we check for other conditions], & we always get only one
 process in CS, so Mutual Exclusion is satisfied.
- ⑧ Checking for progress:-
 Let flag = 0, let execute P_1 1st
- ⑨ P_1 executing 1st instn
 $\therefore R_1 \leftarrow 0$, flag = 1
- ⑩ Assume P_1 is preempted after 1st instn by P_2 ,
 now, P_2 executes 1st instn
- ⑪ $R_2 \leftarrow 1$ & flag = 01
- Now, P_2 executes 2nd instn
 but for that $CMP R_2, \#0$, $ZF \neq 1$ & $ZF = 0$
- ⑫ // P_2 executes 3rd instn & P_2 jumps back to 1st instn, so P_2 is stopped
 by P_1 to enter the CS, though CS is free, but P_1 is interested in
 entering the CS & hence stops P_2 to enter CS.
- ⑬ Now, suppose P_2 preempts P_1 before execution of 4th step (as CS is free)
 but we assume that P_1 will remain in CS till 4th instn gets exec
 ed.
- ⑭ So, progress is satisfied.
 Bounded waiting is not satisfied, because no. of processes are not bounded.

Q. If both the processes P_0 & P_1 are trying to enter into critical section at the same time, then which process will enter critical section first?

- (a) The process which executes statement 2 1st.
 (b) " " " " " 3 1st.
 (c) " " " " " " 4 1st
 (d) We can't say anything, it all depends on situation.

Reason: Initially

- ① $interested[0] = interested[1] = false$.
 ② Now, executing P_0 first.
 $\therefore Other = 1$ & $process = 0$.

In 3rd step, $interested[0] = true$.

& assume P_1 executes 3rd step after it, $\therefore interested[1] = true$.

- ③ Now, for 4th step

let P_1 execute it 1st

$$\therefore turn = 1$$

& now, P_0 execute it

$$\therefore turn = 0.$$

- ④ Now, both checking 5th instn
 Let P_1 execute it 1st

\therefore while ($turn = 1$ & $interested[Other] = true$);
 false, as $turn = 0$.

Cosmos

Test & Set

- Deadlock can't occur.
- But Starvation can occur when there are ~~10 processes~~ ^{10 processes} suppose process P₀ with Test & Set the flag always & hence is always executed, & hence can cause starvation.

So, P₀ is executed 1st because it executes 4th instⁿ.

Using Preemption

→ For P₀ process

- When it executes 3rd instⁿ
interested[0] = true.
- Now, P₁ preempts it.
- So, P₁ executes now
in 3rd instⁿ
interested[1] = true.

Now, P₀ executes 4th instⁿ

∴ turn = 1

Now, P₁ executes 5th instⁿ

but P₁ is stopped at 5th instⁿ.

- Now, P₀ comes back after 3rd instⁿ,
∴ P₀ executes 4th instⁿ
∴ turn = 0
- Now, P₀ executes 5th instⁿ & gets struck there,
- & when P₁ executes 5th instⁿ now, & will get its CS executed.

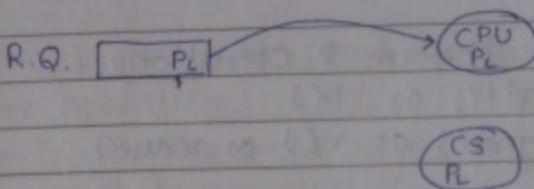
Cosmos

Soln.	M.E.	Progress	Bounded Waiting
1. Lock variable	X	✓	X
2. Strict alternate	✓	X	✓
3. Peterson (but applicable for 2 processes)	✓	✓	✓
4. TSL	✓	✓	X

Cosmos

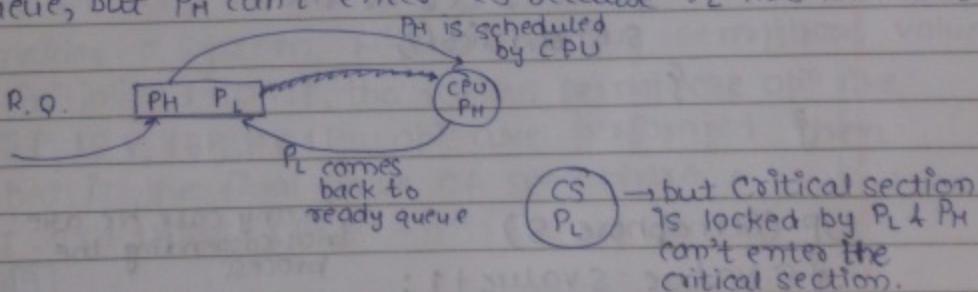
Priority Inversion Problem

1. Assume that we have only one process P_L with low priority in ready queue.



Since, only one process, so it will be scheduled for CPU & P_L will be in the critical section.

2. Now, suppose a high priority process P_H preempts P_L when P_L is executing its CS.
3. So, P_H is scheduled to CPU & P_L will come in ready queue, but P_H can't enter CS because P_L has locked it.

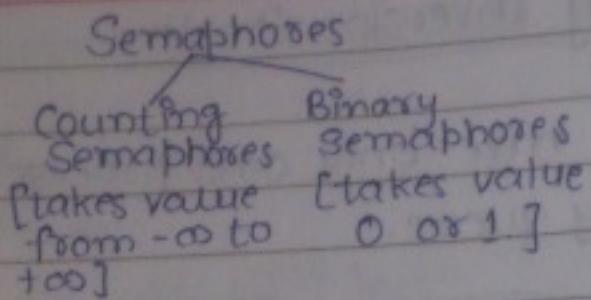


Operating System Soln.

Semaphore :-

- It is an integer variable which is used by two processes in a mutual exclusive manner to achieve synchronisation.
- Improper usage of semaphore will also give improper results.

Cosmos



The Semaphores perform 2 operations:-

1. Down() or Wait() or p()
2. Up() or Signal() or v() or release()

Counting Semaphore :-

1. Down(Semaphore S)

{ S.value = S.value - 1;
 if (S.value < 0)
 { Block the process &
 place its PCB in the
 suspend_list();

g

Up (Semaphore S)

{ S.value = S.value + 1;
 if (S.value <= 0)
 { Select a process from
 Suspend list() & wakeup();

g

In any case, we are
 not suspending the
 process.

the process is
 allowed to perform
 the remaining code
 of down op.

* After performing Down operation, if process is getting suspended, then it is called as unsuccessful down operation.

If down operation is unsuccessful, the process will not continue the execution. [i.e. the remain in down() after

* After performing down operation, if process

Cosmos

classmate

Date _____

Page _____

is not getting suspended, then it is called as successful down operation, if it is successful down opⁿ, it will continue further execution. [i.e. the code after the if block in the down() opⁿ.]

- ★ So, the process will come in suspend-list in unsuccessful down operation.
- ★ There is no unsuccessful up operation, it is always successful.
- ★ Down opⁿ is successful if the semaphore value is greater than equal to 1 (A not 0).
- ★ If Semaphore value is +7, this means we can perform 7 successful down opⁿ.
- ★ If Semaphore value is -7, this means we have 7 suspended processes.

Q. Consider a system where counting semaphore value is initialized to +7, the various semaphore opⁿ like 23P, 18V, 16P, 14V, 1P opⁿ are performed, then what is the final value of semaphore?

Ans. $+7 - 23 + 18 - 16 + 14 - 1 = +9$

the value of semaphore is 9.

Binary Semaphore

Down (Semaphore S)

{ if (S.value == 1)

 S.value = 0;

else

{ Block the process & place its PCB in the suspend list;

 if semaphore value is 1, then we will continue the code after the else block.

 but if it is 0,

 the process will be put into its end list

Cosmos

Up(Semaphore S)

```

{ if (Suspended list() is empty) → if suspended
  { if S.value = 1; list is empty,
    then there are
    no suspended
    processes & hence
    we can't wake
    any process.
    else
      { select a process from
        suspended list & wakeup();
        if it is not
        empty, then
        we will wake
        any process &
        continue the
        execution.
  }
}

```

The down opⁿ is successful only when semaphore value is 1.

There is no unsuccessful up operation.

Q. Each process $i=1$ to g , executes the below code:-

Repeat

p(mutex);

[CS]

forever v(mutex);

The process p_{10} executes below code

Repeat

v(mutex);

[CS]

v(mutex);

forever

The initial value of binary semaphore mutex=1
 what is the max. no. of process that may
 be present inside the critical section at any
 pt. of time?

(a) 2
 ✓ (b) 3

(c) 9
 ✓ (d) 10

Cosmos

P₁, P₁₀, P₂
P₁, P₂, P₃, P₄, P₁₀, P₇

classmate

Date _____

Page _____

My Answer :- (b) 3

Reason:- Suppose 1st process enters

correct:- (d) 10

- Initially P₁ is in CS,
& all the other processes P₂ to P₉ are in suspend list.
- When code for P₁₀ executes
then it executes v(),
∴ P₁, P₂ is in CS
& P₁₀ also comes in CS
∴ P₁, P₂, P₁₀ is in CS.
Now, when P₁₀ comes out of CS, & calls v()
∴ P₁, P₂, P₃ is in CS.
& now code repeats again,
& v() is called again.
∴ P₁, P₂, P₃, P₄ is in CS
& P₁₀ also comes into CS
∴ P₁, P₂, P₃, P₄, P₁₀ are in CS.
∴ similarly all the processes can come into CS.

Q. Now if change P₁₀ code as:-

→ Repeat

v(mutex);

[CS]

p(mutex);

→ forever

→ Now, max. no. of processes in CS are 3.

Q. Now, if we change P₁₀ code as:-

Repeat
p(mutex);
[forever v(mutex);]

then only 1 process can be in CS
at any pt. of time.

Cosmos

Q. Consider two concurrent processes 'P' & 'Q' executes their respective codes

P :-

while (true)

$\int \Sigma$:

~~print('0');~~

25

2

1

while(true)

84: —

```
point(1);
```

```
print('1');
```

Z:

三

- (a) $w = p(T), x = v(T), y = p(S), z = v(S), S = T = 1$
 (b) $w = p(T), x = v(T), y = p(S), z = v(S), S = 1, T = 0$
 (c) $w = p(T), x = v(S), y = p(S), z = v(T), S = T \leq 1$
 (d) $w = p(T), x = v(S), y = p(S), z = v(T), T = 1, S = 0$

5-1
00

* If we have $S=1$, then it can start the o/p as well.
∴ $T=1$ & $S=0$.

Q. Which of the below ensures that o/p string never contain ~~as~~ a substring of 01^n0 (or) 10^n1 where n is odd?

- (a) $w = p(s), x = v(s), y = p(t), z = v(t), s = t = 1$

(b) $w = p(s), x = v(t), y = p(t), z = v(s), s = t = 1$

(c) ~~$w = p(s), x = v(s), y = p(s), z = v(s), s = 1$~~

(d) ~~$w = v(s), x = v(t), y = p(s), z = p(t), s = t = 1$~~

Cosmos

classmate

Date _____

Page _____

-Ans. (a), (b), (d) are pointing 010, (c) is not pointing 010.

Q. Page 45, Q-20.

Case 1: Starting with P_0

$$P_0 \rightarrow P_1 \rightarrow P_0 \rightarrow P_2 \rightarrow P_0$$

$\therefore O/p :- 000$

Case 2: $P_0 \xleftarrow{P_1} P_0 \xrightarrow{P_2} P_0$ (when both P_1 &
DO P_2 are executed
parallelly)

$$S_0 = X \otimes X X \otimes 0$$
$$S_1 = \varnothing \otimes \varnothing 1$$
$$S_2 = \varnothing X \otimes 1$$

Classical problems

Producer-Consumer:-

Semaphore mutex = 1;

Semaphore empty = N;

Semaphore full = 0;

void producer(void)

{ int Itemp;

while(true)

{ produce_item(Itemp);

down(empty);

down(mutex);

Buffer[in] = Itemp;

in = (in+1) Mod N;

up(mutex);

up(empty/full);

g

g

- * Mutex is a binary semaphore variable used by the producer & consumer to access the buffer in a mutual exclusive manner.
- * Empty is a counting semaphore variable representing the no. of empty slots in the buffer at any point of time.
- * Full - a counting semaphore variable representing no. of filled slots in the buffer at any point of time.

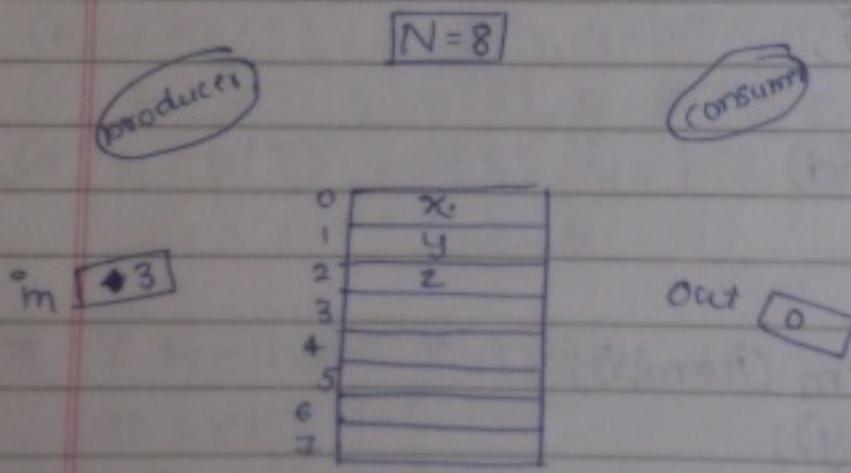
void consumer (void)

```

{ int Itemc;
  while(true)
  { down (full);
    down (mutex);
    Itemc = Buffer [out];
    out = (out + 1) % mod N;
    up (mutex);
    up (empty);
    process_item (itemc);
  }
}
  
```

Cosmos

Analysis



In this example:-

- empty = 5
 - + full = 3
 - we are producing an item
- down(empty)
- \therefore empty = 4
- ↓ down(mutex)
- \therefore mutex = 0.

Cosmos

classmate

Date _____

Page _____

→ Now,

$$in = (n+1) \bmod N$$

$$\therefore in = 4.$$

→ Now, up(mutex)

$$\therefore mutex = 1$$

& up(full)

$$\therefore full = full + 1;$$

$$\therefore full = 4.$$

→ Now, we execute consumer program

down(full)

$$\therefore full = 3$$

& down(mutex)

$$\therefore mutex = 0$$

→ Now,

$$out = (out + 1) \bmod N$$

$$\therefore out = 1$$

→ Now, up(mutex) $\therefore mutex = 0$

→ & now, up(empty), $\therefore empty = 4$

∴ now,

* If Buffer is full

$$Empty = 0$$

$$Full = 8$$

so, producer is not allowed to produce by statement

down(empty)

\therefore the producer will halt here.

* If Buffer is empty

$$Empty = 8$$

$$Full = 0$$

so consumer is not allowed to consume by

down(full).

\therefore consumer will halt.

Cosmos

- Q. If we interchange down(empty) & down(mutex), then
- We encounter down(mutex) 1st which will be executed, & goes mutex=0, & goes to next instn.
 - down(empty) if empty=0, then producer halts here.
 - The consumer will now execute down(full) which makes the exec. of next instn which is down(mutex) but mutex=0, so consumer also halts,
 - So, both producer & consumer halts & hence deadlock.

Readers-Writers Problem:-

int $rc = 0$; → reader's count :- it represents the no. of readers present in the db/d (database) at any point of time.

Semaphore mutex=1; → it is a binary semaphore used by the readers.

Semaphore db=1; → it is also a binary semaphores used by readers & writers.

void Reader(void)

{ while(TRUE)

 { down(mutex);

$rc = rc + 1$;

 if ($rc == 1$) down(db);

 up(mutex);

 [DB]

 down(mutex);

$rc = rc - 1$;

 if ($rc == 0$) up(db);

 } up(mutex);

}

```

void writer(void)
{
    while(TRUE)
    {
        down(db);
        DB
        up(db);
    }
}

```

Cosmos

If reader is in the database, then writer is not allowed to update the database.

If reader is in the database, then readers are allowed.

If writer is in the d/b, then readers are not allowed

Analysis

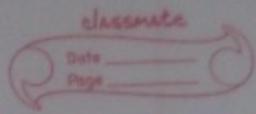
$$rc = 0, mutex = 1, db = 1$$

- ① Assume 1st reader is entering into d/b.
- ② down(mutex)
- ③ Now, $rc = rc + 1 = 1$
- ④ Now, checking $rc == 1$, which is true,
 \therefore down(db), $\therefore db = 1$
- ⑤ Now, up(mutex), $\therefore mutex = 1$
- ⑥ Now, 1st reader is in the db.
- ⑦ Now, we will check whether writer can come in the d/b or not.
- ⑧ So, we will execute writer code.
- ⑨ So, writer has down(db)
 say but $db = 0$, & hence writer is suspended.

Cosmos

- (10) Now, we will check whether 2nd reader can enter the d/b.
- (11) So, we execute reader's code.
 down(mutex)
 $\therefore \text{mutex} = 0,$
 $\& \text{rc} = \text{rc} + 1 = 2$
 $\& (\text{rc} == 1) \text{ is false}$
- (12) up(mutex)
 $\text{mutex} = 1$
 $\& \text{hence the 2nd reader enters the d/b}$
 $\text{which is acc. to the problem.}$
-
- (1) Now, 1st writer enters the d/b, so writer's code executes.
- (2) $\therefore \text{down(db)}$
 $\therefore \text{db} = 0$
 $\& \text{writer is in the d/b.}$
- (3) Now, we will check while writer is in db, is the reader allowed? so we execute reader's code.
- (4) down(mutex)
 $\therefore \text{mutex} = 0$
 $\& \text{rc} = \text{rc} + 1 = 1$
- (5) $\& (\text{rc} == 1) \text{ is true}$
 $\& \text{down(db)}$
[since $\text{db} = 0$ prior to down(db) , \therefore the reader will be suspended.]

Cosmos



Q. `int R=0, W=0;`

`Semaphore mutex=1;`

`void Reader(void)`

{ `L1: down(mutex);`

`if(W==1)`

 { `[] → ① up(mutex)`

`goto L1;`

`else`

 { `R = R + 1;`

`[] → ② up(mutex)`

`|`

`DB`

`down(mutex);`

`R = R - 1;`

`up(mutex);`

`|`

`void writer(void)`

{ `L2: down(mutex);`

`if([] → ③ R ≥ 1 or W == 1)`

`up(mutex);`

`goto L2;`

`W = 1;`

`up(mutex);`

`DB`

`down(mutex);`

`W = 0;`

`up(mutex);`

`|`

Q. What should be the values of blanks 1,2,3 to synchronise classical readers & writers?

(a) `down(mutex), up(mutex), W == 1`

(b) `up(mutex), up(mutex), W == 1`

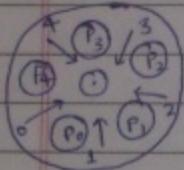
(c) `down(mutex), up(mutex), R ≥ 1 or W == 1`

(d) `up(mutex), up(mutex), R ≥ 1 W == 1`

Cosmos

- Q. What happens if we interchange `w=1 & up(mutex)` in writer's code.
- (a) Soln works fine, no problem at all.
- (b) Only multiple writers are allowed.
- (c) readers & writers both can be in DB at same time.
- (d) None of them.

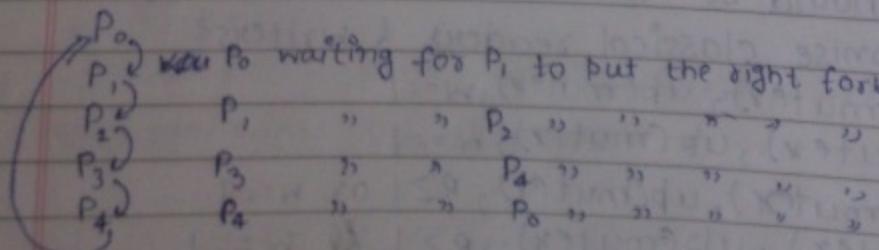
Dining Philosophers



```

void philosopher(int i)
{
    while(true)
        thinking(); → pick the left
        take-fork(i); → fork
        take-fork((i+1)%N); → pick the right fork
        eat();
        put-fork(i); → put the left
                           fork back on table.
        put-fork((i+1)%N); → put the right
                           fork back on table.
}

```



So, it's a deadlock.

Cosmos

void test (int i)

{ if (state[i] == HUNGRY && state[LEFT] != EATING &&

state[RIGHT] != EATING)

{ state[i] = EATING;

up(sem);

}

}

Q. Let $P[0] \dots P[4]$ be processes & $M[0] \dots M[4]$

be binary semaphores, mutex variable is initialised to 1.

Each process P_i executes the below code.

$0 \rightarrow 1$
 $1 \rightarrow 2$
 $2 \rightarrow 3$
 $3 \rightarrow 4$
 $4 \rightarrow 0$

wait ($m[i]$);
 wait ($m[(i+1) \bmod 4]$);

[CS]

signal ($m[i]$);
 signal ($m[(i+1) \bmod 4]$);

0 1 2 3 4
 1 2 3 4
 0 1 2 3 4
 3 1 2 0 4

Consider the statements below:-

I. M.E. is satisfied

II. M.E. is not satisfied

III. It is possible for deadlock

Which of the above statements are true

(a) only I

(b) only I & III

✓ (c) only II & III

(d) only II

★ Deadlock :- When all the processes tries to enter [CS], then 1st wait statement is successful, but 2nd wait opn causes all processes to be in suspended state.

Date _____

24.06.12

classmate

Date _____
Page _____

Cosmos

Concurrent Programming

- It is possible when we have multi-processors system.

★

$S_1: a = b + c;$

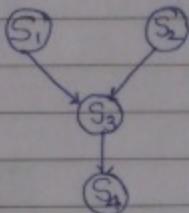
S_1 & S_2 are independent statements

$S_2: d = e * f;$

$S_3: g = a / d;$

$S_4: h = g * i;$

Dependency Graph



S_1 & S_2 can be run concurrently (i.e. independently) or parallelly.

Read Set: {b, c, e, f, a, d, g, i}

Write Set: {a, d, g, h}

S_i & S_j can be run concurrently or parallelly :-

- ① $R(S_i) \cap W(S_j) = \emptyset$ R : read set
- ② $R(W(S_i)) \cap R(S_j) = \emptyset$ W : write set
- ③ $W(S_i) \cap W(S_j) = \emptyset$

To write programs parallelly:
pair begin (or) co-begin
pair end (or) co-end

If we write:-

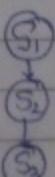
begin

$S_1:$

$S_2:$

$S_3:$

end



If we write:-

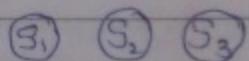
pair begin

$S_1:$

$S_2:$

$S_3:$

pair end



Cosmos

example: $S_0;$

par begin

begin

$S_1;$

$S_2;$

end

begin

$S_3;$

$S_4;$

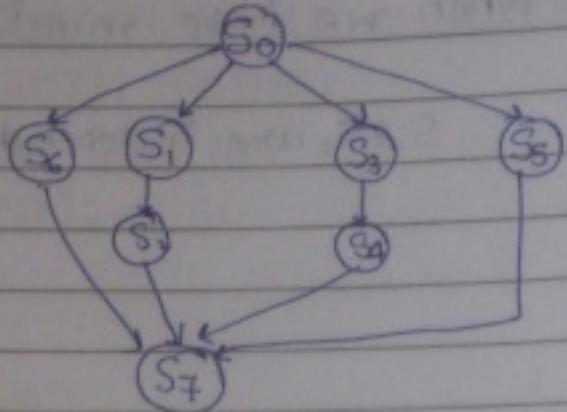
end

$S_5;$

$S_6;$

par end

$S_7;$



Q. Dependency Graph begin

$S_1;$

par begin

begin

$S_2;$

$S_4;$

par begin

$S_5;$

$S_6;$

par end

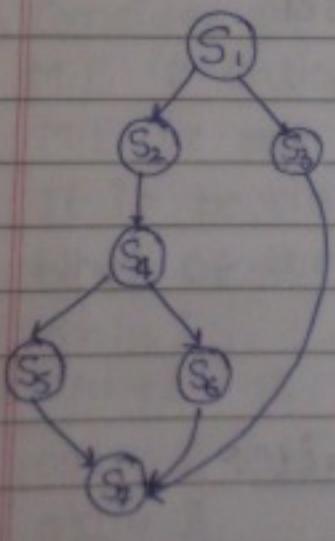
end

$S_3;$

par end

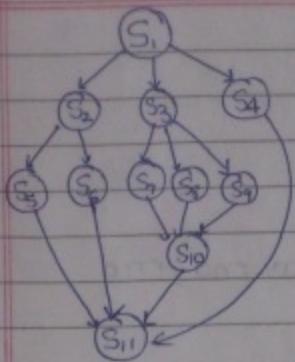
$S_7;$

end



Cosmos

Q.



begin

S₁;

par begin

begin

S₂;

par begin

S₅;S₆;

par end

end

begin

S₃;

par begin

S₇;S₈;S₉;

par end

S₁₀;

end

S₁₀; par end;S₁₁;

end

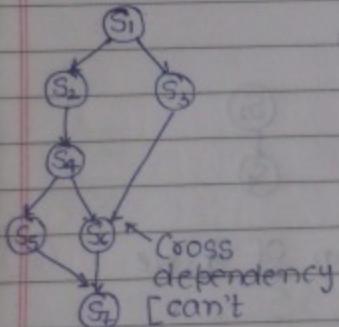
begin

S₁;

par begin

S₃;

end

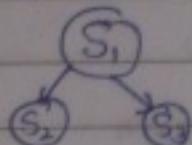


Cosmos

begin $p(d), p(f), S_6, v(h)$, end
 begin $p(g), p(h), S_2$, end
 par end

→ In 1st instn

- begin S_1 ; par begin $v(a), v(b)$, par end, end

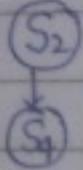


~~Wt~~ S_2 & S_3 can only be executed when S_1 is completed.

& S_2 & S_3 are in parallel,

∴ we perform $v(a)$ & $v(b)$
 for S_2 for S_3

- begin ~~p(a)~~ $p(a), S_2, v(c)$, end
 \downarrow
 for S_4

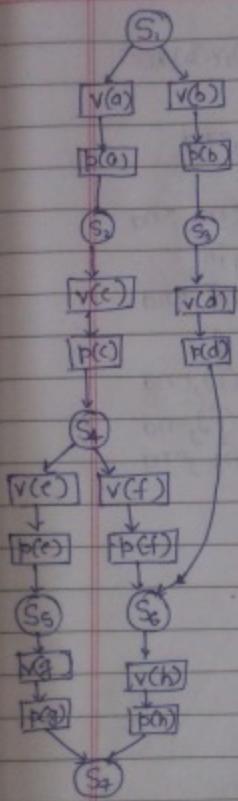


$p(a)$ → means only after completion of S_1 , we can execute S_2 .

$v(c)$ → we are upping semaphore c , which means S_4 can only be executed when S_2 is executed.

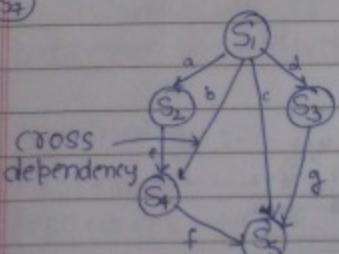
- & so on....





Cosmos

Q.



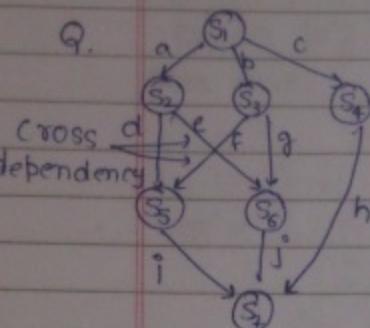
has begin

begin S₁: has begin v(a), v(b), v(e), v(d),
has end, end dbegin : p(a), S₂, v(e), endbegin : p(b), S₃begin : p(c), p(b), S₄, v(f), endbegin : p(d), S₃, v(g), endbegin : p(c), p(g), p(f), S₅, end

has end

Cosmos

Q.



par begin

begin S₁: bar begin v(a), v(b), v(c),
bar end, endbegin : b(a), S₂, v(d), end

begin : \$(), bar begin

v(a), v(e), bar end, end

begin : p(b), S₃, bar begin

v(f), v(g), bar end, end

begin : b(c), S₄, v(h), endbegin : b(d), b(f), S₅, v(i), endbegin : b(g), b(e), S₆, v(j), endbegin : p(i), p(j), b(h), S₇, end
bar end

Q. int x=0, y=0;

par begin

begin

x=1;

y = y + x;

end

x+1	y= 2
y=L	x= 5

x=1

y

x=1

y=2

y=5

x=4

par end

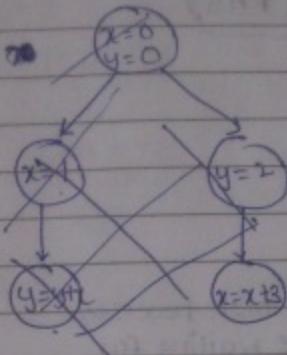
What are the possible values of x & y after completion of program

I x=1, y=2

II x=1, y=3 ✓

III x=4, y=6 ✓

Ans. II & III



Cosmos

II \rightarrow $y = 2$
 $x = x + 3$
 $x = 1$
 $y = y + x$

x y
 1 3

II \rightarrow $x = 1$
 $y = 2$
 $x = x + 3$
 $y = y + x$

x y
 4 6

Deadlock

Definition :-

If two or more processes are waiting on some event to happen which never happens, then we say those processes are involved in deadlock.

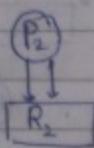
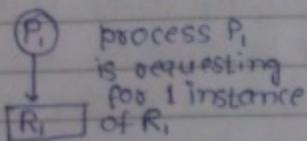
Terminology :-

① Processes \rightarrow P_1 P_2

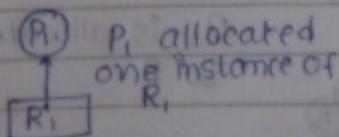
② Resources \rightarrow R_1 R_2

$R_1, R_2 \rightarrow$ means 2 instances of R_1 , R_2

③ Requesting



④ Allocation

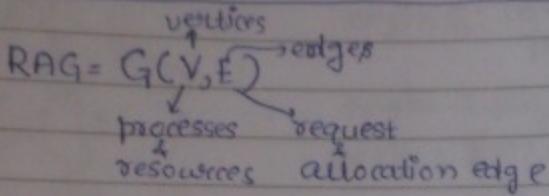


Cosmos

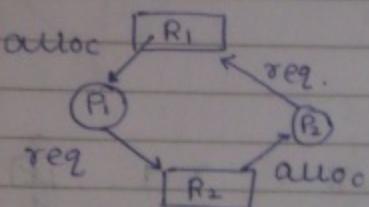
classmate

Date _____
Page _____

Resource Allocation Graph (RAG)

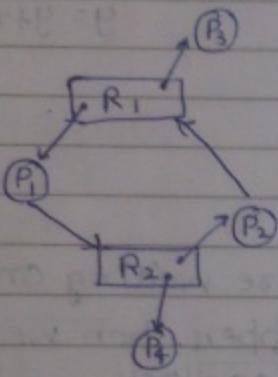


example:-



1. Two or more processes
2. They are waiting for some event to happen
3. Event never happens

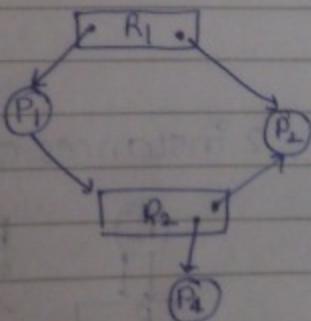
Deadlock exist in RAG.



1. Two or more processes
2. P1 & P2 are waiting for R1 & R3 respectively.
3. But R1 & R2 can be allocated to P1 & P2 respectively at some pt. of time.

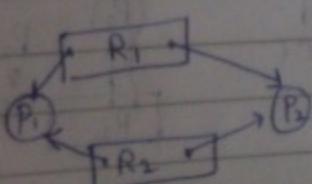
So no deadlock.

When P3 executes



R1 is given to P1

When P4 executes



R2 is allocated to P1.

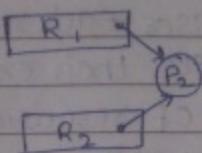
Cosmos

classmate

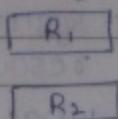
Date _____

Page _____

When P_1 executes



When P_2 executes



Life-Cycle of Resource Request & Release :-

1. The process request for the resource.
2. O.S. validates the request of the process.
(If the resource is not available for the system,
i.e. if there is no pointer & a process is requesting
pointer, then it is an invalid request & is rejected
by OS.)
3. O.S. checks the availability of the resource.
4. If the resource is available, it will be allocated to
the process, otherwise process has to wait.
5. The process will go into execution only if all the
resources required by the process at that pt. of time
are allocated.
6. The process releases all the resources after completion
of the process execution.

→ Read

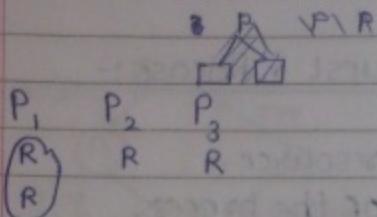
- Q. Consider a system which has n processes & 6 tape drives. If each process requires 2 tape drives to complete their execution, then what is the max. value of n which ensures deadlock free operation?

- (a) 2
(b) 3
(c) 4
(d) 5

Cosmos

classmate
Date _____
Page _____

- Q. Consider a system with 3 processes each require 2 units of resource R to complete their execution. Then what is the min. no. of resources required to ensure deadlock free opn.



Ans. 4 resources.

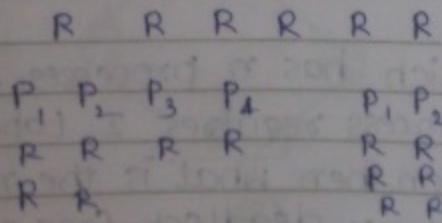
2 can't be the right answer, because

P₁ P₂ P₃ → deadlock
R R R

3 can't be the right answer, because

P₁ P₂ P₃ → deadlock
R R R

- Q. Consider a system with n processes & 6 tape drives. If each process requires 3 tape drives to complete their execution, then what is the max. value of n for deadlock free opn.



Ans. if we have max. 4 processes:

P₁ P₂ P₃ P₄ → deadlock
R R R R
R R

If we have 3 processes:-

$P_1 P_2 P_3$
R R R → deadlock
R R R

Cosmos

If we have 2 processes:-

$P_1 P_2$
R R → no deadlock
R R
R R

∴ ans. is ②.

Q. Consider a system with 3 processes P_1, P_2, P_3 , the peak demand of each process is 5, 9, 13 respect. for resource R. Then what is the min. no. of resources req. for deadlock free opn.

- (a) 22
- (b) 23
- (c) 24
- (d) 25

P_1	P_2	P_3
5	9	13
0	8	15

$$P_1 \rightarrow 5 \leftarrow$$

$$P_2 \rightarrow 9 \leftarrow$$

$$P_3 \rightarrow 13 \leftarrow$$

If we give 4 to P_1 & 8 to P_2 & 12 to P_3 , then

$$4 + 8 + 12 = 24$$

It can cause deadlock

So, if we give 1 additional resource, then there will be no deadlock

$$\therefore 24 + 1 = 25$$

Cosmos

classmate

Date _____
Page _____

Page
48 Q11

$P_1 \ P_2 \ P_3 \ P_4$

4 3 7 8

→ 20 identical resources

9 9 9
9 9 9
9 9 9

$P_1 \rightarrow 4$

total is 20.

$P_2 \rightarrow 3$

$P_3 \rightarrow 7$

$P_4 \rightarrow ?$

$$3 + 2 + 7 = 12$$

$$20 - 12 = 8$$

$$P_1 \rightarrow 4 \rightarrow 3$$

$$P_2 \rightarrow 3 \rightarrow 2$$

$$P_3 \rightarrow 7 \rightarrow 6$$

$$P_4 \rightarrow ? \rightarrow 8$$

$$\frac{12}{19} + 1, \text{ additional resource for deadlock free opn}$$

Page
49 Q18

S1°

$P_1 \ P_2 \ P_3 \dots \ P_n$

1 1 1 1

Sufficient condition :-

least upper bound which satisfy the condition.

$P_1 \ P_2 \ P_3 \ P_4 \dots \ P_m$

$\forall S_i < m$

$m=10$

$n=100$

$P_1 \rightarrow 9$

$P_2 \rightarrow 9$

$P_3 \rightarrow 9$

⋮

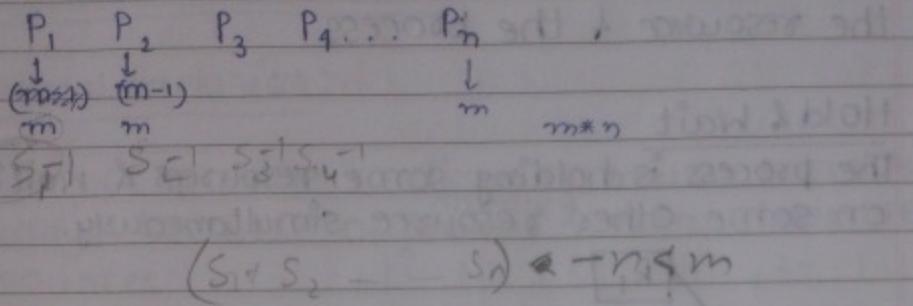
$P_{100} \rightarrow 9$

but $m=10$ $\frac{900}{10}$
↳ deadlock

$\forall i; S_i < n$

↓
no meaning at all

Cosmos



$$(S_1 - 1) + (S_2 - 1) + \dots + (S_n - 1) < P_1 + P_2 + P_3 + P_4 + \dots + P_n \rightarrow \text{for deadlock}$$

Q15
Page 9

$n \rightarrow$ process

$i \rightarrow$ holding x_i of R

$$y_p = y_q = 0$$

$$x_0 + x_1 + \dots + x_{n-1} = R$$

$(n-2)$ → are competing

$$\min(x_p, x_q) < \max y_k$$

Deadlock Characteristics:-

1. Mutual Exclusion
 2. Hold & Wait
 3. No pre-emption
 4. Circular Wait

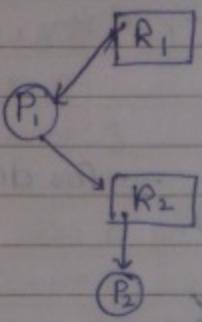
Cosmos

Mutual Exclusion

- The resource has to be allocated to only one process or it is freely available.
- There should be a 1-to-1 relationship b/w the resource & the process.

Hold & Wait

- The process is holding some resources & waiting on some other resource simultaneously.

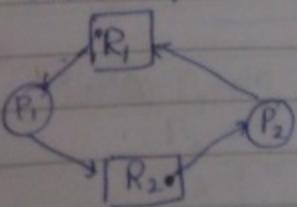


No pre-emption

- The resource has to be voluntarily released by the process after the completion of execution.
- It is not allowed to pre-empt the resource forcefully from the process.

Circular Wait

- The processes are circularly waiting on each other for the resources.



* If only all the processes above are occurring simultaneously in the system, then

Cosmos

classmate

Date _____
Page _____

definitely there exist a deadlock.

Deadlock Prevention

1. Mutual Exclusion:-

It is not possible to dissatisfy mutual exclusion always because of ~~shared~~^{shareable} & non-shareable resources.

- File is shareable resource
it can be read by two or more processes.
- but Printer is Non-Shareable resource.

2. Hold & Wait :-

(i) Allocate all the resources required by the process before the start of execution.

If we follow this strategy we will have 100% device utilization.

(ii) The process should release all the existing resources before making a new request.

If we follow this strategy, then it is possible for starvation.

(The process is not under execution).

3. No pre-emption:-

Process P_1 requesting resource R_1 ,

if R_1 is available,
then it will be allocated to P_1 .

if R_1 is busy, & it is allocated to other process P_2 , then

if P_2 is in execution, then
 P_1 will wait.

if P_2 is not in execution & it is waiting for other resource R_2 , then we will pre-empt P_2 & take R_1 from it & allocate it to P_1 .

Cosmos

4. Circular Wait

Every resource will be assigned with a numerical no. The process can request for the resource only in the increasing order of enumeration.

e.g.

$P_1 \rightarrow R_7$	$P_1 \rightarrow R_5$	$P_1 \rightarrow R_9$
granted	rejected	granted

Deadlock Avoidance

Banker's Algorithm

	Maximum need	Current Allocation	Current Available	Request
--	--------------	--------------------	-------------------	---------

Total:	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	5	3	0	1	0	3	3	2	7	4	
P_1	3	2	2	2	0	0				1		
P_2	9	0	2	3	0	2				6	0	
P_3	2	2	2	2	1	1				0	1	
P_4	4	5	3	0	0	2				4	3	
				7	2	5						

Remaining Need = Maximum Need - Current Allocation

Current Available = Total Available - Current Allocation

★ Now, we need to identify whether system is in safe state or unsafe state, if it is unsafe state then deadlock can occur. (only possibility) may or may not occur.

Cosmos

classmate

Date _____
Page _____

★ We can multiple safe sequences.

- ① If we can satisfy the remaining need of all the processes with current available resources in some order then the system is said to be in safe state, otherwise system is in unsafe state.
- ② If system is in unsafe state, then deadlock may occur.
- ③ The order in which we satisfy the remaining need of all the processes is called as safe sequence.
- ④ The safe sequence can't be unique, we can have multiple safe sequences.
- ⑤ The unsafe state purely depends on behaviour of the processes.
- ⑥ Deadlock avoidance is more restrictive than deadlock prevention.

	3	5	4	3
P ₃	7	4	3	
P ₁	7	5	3	
P ₀	9	5	5	
P ₂				
P ₄				

ns. → P₁

now available resources =

current allocation + current available

$$= 2,0,0 + 3,3,2$$

$$= 5,3,2$$

$$\begin{array}{r} 2 \ 1 \ 0 \\ 1 \ 2 \ 2 \\ \hline 3 \ 3 \ 2 \end{array}$$

Now P₃

available resources = 5,3,2 + 2,1,1 = 7,4,3

Now P₀

available resources = 7,4,3 + 0,1,0 = 7,5,3

Now P₂

available resources = 7,5,3 + 3,0,2 = 10,5,5

Cosmos

classmate

Date _____
Page _____

Now P_4
 available resources = $10, 5, 5 + 0, 0, 2 = 10, 5, 7$
 \therefore Safe Sequence $\rightarrow [P_1, P_3, P_0, P_2, P_4]$

Page 47

Q3.

P: P_1, P_0, P_2, P_3

$P_1 \rightarrow$ available

now $4, 3, 1 + 2, 1, 2$
 $= 6, 4, 3$

Need

A	B	C	
P_0	6	2	0
P_1	1	3	0
P_2	1	0	2
P_3	2	0	4

$\underline{6, 4, 3}$

$P_0 \rightarrow$ available

now

A	B	C	
P_0	1	3	0
P_1	1	0	2
P_2	2	0	4

$\underline{6, 4, 3}$

Q: P_1, P_2, P_3, P_0

$P_1 \rightarrow$ available

now $4, 3, 1 + 2, 1, 2$
 $= 6, 4, 3$

P_0
 $\underline{6, 4, 3}$
 $0, 3, 4$

P_1
 $\underline{6, 7, 7}$
 $6, 7, 7$

P_2
 $\underline{6, 7, 7}$
 $0, 0, 2$

$\underline{6, 7, 9}$

$P_2 \rightarrow$ available

now $6, 4, 3 + 0, 0, 2$
 $= 6, 4, 5$

P_3

$P_3 \rightarrow$ available

now $6, 4, 5$

P_3
 $\underline{4, 3, 1}$
 $2, 1, 2$
 $\underline{6, 3, 5}$

Q: $\because P_1, P_2, P_3, P_0$

R: $\bullet 4, 3, 1 \quad P_1 \quad P_3$

$\underline{2, 1, 2}$
 $\underline{6, 4, 3}$

$\frac{2}{3}$
 $\frac{3}{5}$
 $\frac{5}{10}$

$P_1 \quad P_2 \quad P_3 \quad P_4 \quad P_5 \quad P_6$

$1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1$

$1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1$

$1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1$

Cosmos

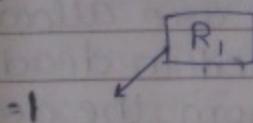
classmate

Date _____
Page _____

2
3
5
10

Deadlock Detection

1. If all the resources are of single instance type
- If Cycle exists in RAG, then deadlock exists



- If all the resources are single instance, then cycle in RAG is necessary & sufficient cond'n for occurrence of deadlock.
- If all the resources aren't single instance, then cycle in RAG is necessary but not sufficient cond'n for occurrence of deadlock.
- If the resources are of multiple instance type, then the Banker's algo is used whether we can satisfy the remaining need of all the processes with current available resources.
- If we can satisfy, we can say that the deadlock doesn't exist, otherwise it exists in the system.

Deadlock Recovery :-

1. Killing the process.
 - Kill all the processes involved in the deadlock.
 - Kill one after the other. Criteria:-
 - (a) Priority (generally low priority process is selected first to be killed.)
 - (b) % of process completion :-
(the process which is more completed is not killed)
 - (c) no. of resources the process is holding
(kill those processes which is holding more no. of resources.)

Cosmos

2. Resource Pre-emption:-

The resource will be pre-empted from the process involved in deadlock.

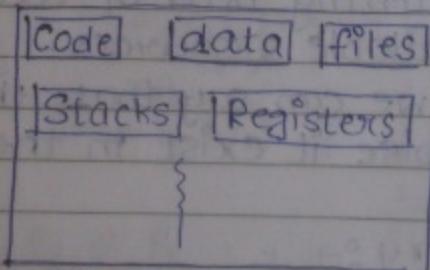
The pre-empted resources are allocated to other processes involved in deadlock to recover the system from the deadlock.

3. Ostrich Algorithm

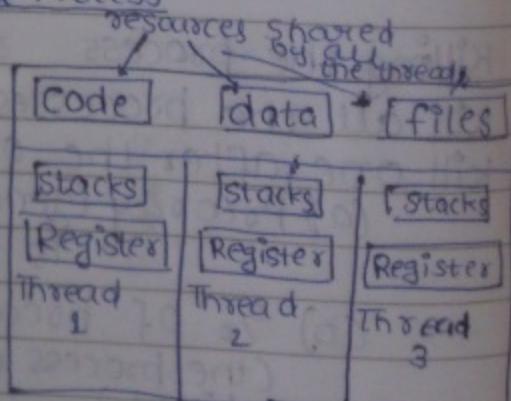
→ Ignore the deadlock.

Threads

efn :- Light weight processes
↳ less context



Single Threaded Process



Multithreaded process

Cosmos

classmate

Date _____

Page _____

Advantages Of Thread :-

1. **Responsiveness:** - The thread will improve the responsiveness. If one thread has completed the execution, the o/p will be responded early as compared to the completion of the process.
2. **Less Context Switching time:** - The thread has less context, ∴ less context switching time.
The context switching time will be less in threads compared to the process.
3. **Resource Sharing:** - The resources of the process will be shared among all the threads within the process.
4. **Effective utilization of multiprocessor systems:-**
If multiprocessor system is used then diff. threads of a process can be scheduled onto diff. processors, so that the process execution will be faster.
5. Implementing the threads is very economical. There are various programming languages which supports implementing the thread. e.g. Java
6. Enhanced throughput of the system:- If the thread is considered as a job, then the no. of jobs completed / unit time increase, hence throughput of the system is enhanced.

Threads → User-level Threads
→ Kernel-level Threads

- The OS can't recognize User-level threads

Cosmos

User level Threads

1. The OS can't recognise user-level threads.

The OS views user-level thread as a process only.

(if one user level thread wants to perform I/O op, then the entire process is moved into wait state)

2. These threads are dependent threads.
3. Less context
4. Doesn't require h/w support.

Kernel level Threads

1. These threads are created by the OS & recognised by the OS.

(if one kernel level thread is performing blocking system call, then another thread will continue the execution)

2. Kernel level are defined as independent threads.

3. More Context
4. They require h/w support.

William Stallings

Memory Management

Main Memory / Physical Mem. / Primary Mem. / RAM

Functionality :- Allocation & Deallocation of memory to the processes.

Goal :- The efficient utilization of the memory by minimizing internal & external fragmentation

$$2^{10} \approx 10^3 \rightarrow 1\text{ K}$$

$$2^{20} \approx 10^6 \rightarrow 1\text{ M}$$

$$2^{30} \approx 10^9 \rightarrow 1\text{ G}$$

$$2^{40} \approx 10^{12} \rightarrow 1\text{ T}$$

Memory → Byte Addressable
Word Addressable.

Cosmos

classmate

Date _____

Page _____

Memory		→ 16 bits	Capacity of memory = 8×16 bits = 8×16 bytes = 8 bytes
0	000	8 Words	
1	001		
2	010		
3	011	each	
4	100		
5	101		
6	110		
7	111		

Capacity of memory = no. of words in memory × word size

Q. Consider a system which has 128 K words, where each word is having size of 64 bits, then what is the capacity of memory in Bytes?

Ans. Capacity = $128 \text{ K} \times 8 \text{ Bytes}$
 $= 1024 \text{ KBytes } [2^{10} \times 2^{10} \text{ Bytes}]$
 $= 1 \text{ M Bytes } [2^{20} \text{ Bytes}]$

$$2^9 \times 2^{10} \times 2^4$$

Q. Consider a system having 512 M words, each word is having size of 16 bytes

Ans. $2^9 \times 2^{20} \times 2^4 \text{ Bytes}$
 $= 2^{30} \times 2^3 \text{ Bytes } \rightarrow 1 \text{ G Bytes}$
 $= 8 \text{ G Bytes } [8 \times 2^{30} \text{ Bytes}]$

Q. Consider a system where 25 bits are used to represent the words of the mem., each word has the size of 32 bits

Ans. $= 2^{25} \times 4 \text{ Bytes}$
 $= 2^7 \text{ Bytes}$
 $= 2^7 \times 2^{20} \text{ Bytes}$
 $= 128 \text{ M Bytes}$

Cosmos

RAM chip implementation

RAM chip size = 128 Bytes

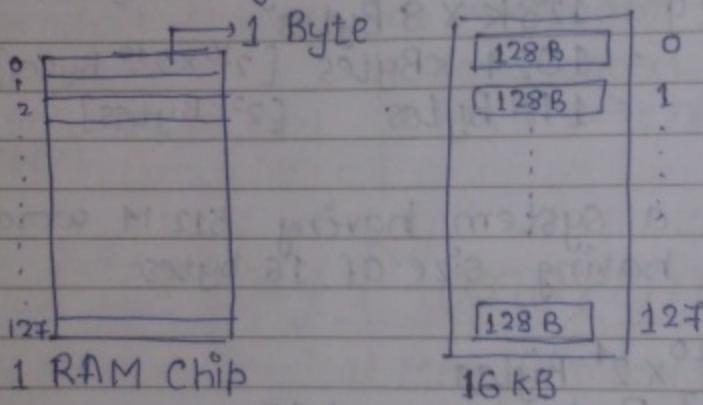
RAM chip size - 128 Bytes
organize the mem. capacity of 16 KB

$$\frac{2^4 \times 2^{10}}{2^4 \cdot 2^3} = 2^7 = 128$$

(a) How many RAM chips are req.

$$\frac{2^4 \times 2^{10} \text{ bits}}{2^7 \text{ bits}} = \boxed{128} \text{ RAM chips}$$

(b) Draw Mem. org. map!

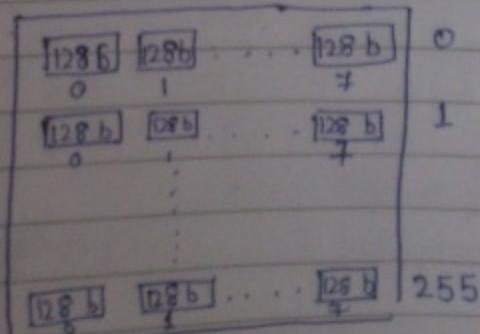


RAM chip size = $(128 \times 1\text{-bit})$ → each word of 1 bit

Mem. cap. = $32 \text{ K} \times B$ → each word of 1 byte
→ 32 K words

$$\frac{2^5 \times 2^{10} \times 2^3}{2^7} = \frac{2^{18}}{2^7} = 2^{11} = 2048$$

RAM
chips



Cosmos

classmate

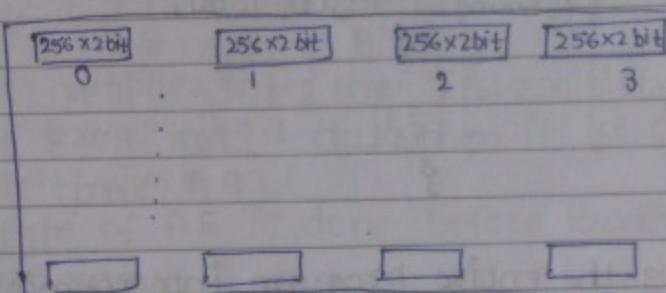
Date _____

Page _____

Q. RAM chip size = $(256 \times 2 \text{ bit})$ → 256 words
 Capacity = (64 MB) → each word of 1 Byte each
 L 64 M words

$$\text{No. of RAM chips} = \frac{2^6 \times 2^{20} \times 2^3}{2^8 \times 2} = \frac{2^{29}}{2^9} = 2^{20}$$

1 Million RAM chips.



~~64 2²²~~

$$2^{26} B$$

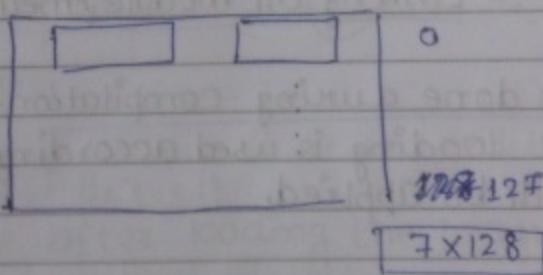
$$2^{18} \times 2^8 = 2^{26} B$$

$$2^{18}-1$$

Page no.

54
Q38.

$$\text{no. of RAM chips} = \frac{2^5 \times 2^{10} \times 2^3}{2^8 \times 4} = \frac{2^{18}}{2^8} = 2^8$$



Cosmos

0.06.12

Loading & Linking :-

Static dynamic

```
main()
{
    int x;
    x = add();
    f();
}
```

int add()
 {
 int total;
 return total;
 }

linking

- Loading the entire program into memory before the starting of execution.
- Static loading suffers from inefficient utilization of memory.
- The program execution will be faster.

Linking :- Linking all modules/methods of the program.

- Linking is done during compilation.
- If static loading is used accordingly, the static linking will be applied.

Dynamic:-

- Loading the program into memory on demand (whenever it is required.)
- Efficient utilization of memory.
- Program execution will be slower.
- If the Dynamic loading is used acc., the dynamic linking will be applied.

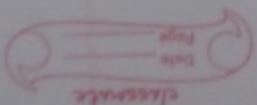
- The majority of the OS uses dynamic loading & dynamic linking concept.
- Address Binding:-
- Physical mem. locations is called as address binding.
- Association of program instruction & data to the actual memory:-
- If the compiler takes the responsibility of association of address binding:-
- Compiler time Address Binding:-
- There are 3 types of address binding:-
- Physical time Address Binding:-
- This type of A.B. is done before loading the prog.
- The compiler needs to interact with O.S memory into memory.
- Program to perform compile time A.B.
- Memory

0
1
2
3
4

P_i → I₁ ← I₁₀ ; - I₁ first is stored at 10. mem address of 10.

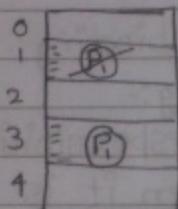
P_i → I₁ ← I₂ ← I₃ ← I₄ ← I₅ ← I₆ ← I₇ ← I₈ ← I₉ ← I₁₀ ; - I₁ first is stored at 10. mem address of 10.

COSMOS



Cosmos

- This is done at the time of exec.
- The A.B. is done by processor (or CPU).
- Majority of the OS follows Run time or Dynamic A.B.
for run time



When the program is moved from block 1 to block 3 during run time due to paging, etc., the corresponding to block 1 we have diff. binded addresses of the inst? than from block 3,
∴ A.B. is done during run time.

Memory Management Techniques:-

Contiguous

- fixed partition scheme
- variable partition scheme

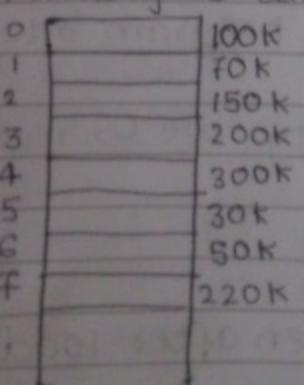
Non-contiguous

- paging
- multi-level paging
- Inverted paging
- segmentation
- Segmented paging

Contiguous :-

Fixed partition Scheme :-

- Memory is divided into fixed no. of partitions.



- No. of partitions are fixed, but not their size.

e.g. we have 8 partitions here, which are fixed.

- In every partition, we can only store 1 process,
- ∴ no. of processes are restricted by no. of partitions in the memory. (or degree of multiprog. is restricted by no. of partitions)

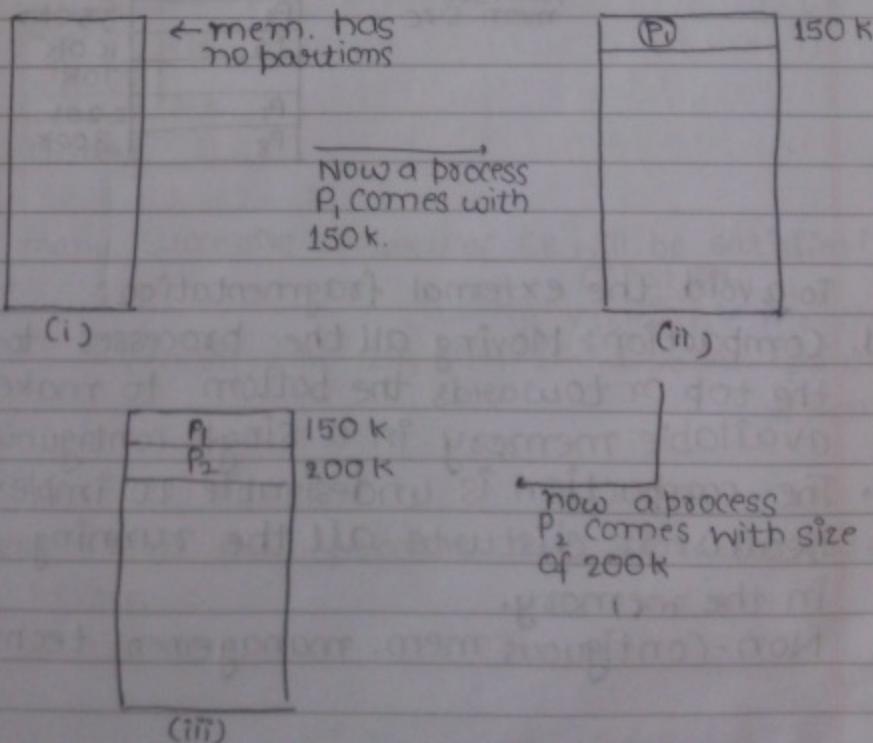
Cosmos

- The max. size of the process is restricted by max. size of the partition.
e.g. we can't place 350k process in our example memory.
- If we accommodate 20k process in partition of 30k, then 10k is wasted. This wastage of memory is called Internal fragmentation.
- Every partition is associated with limit register, so lower limit contains starting address of the partition & upper limit contains ending address of partition.

Variable Partition Scheme

- In variable Partition scheme, initially the memory will be full contiguous free block.

So initially:-



Cosmos

P ₁	150k	P ₁	150k
P ₂	200k	P ₂	200k
P ₃	50k	P ₃	50k
P ₄	350k	P ₄	350k
P ₅	100k	P ₅	100k
P ₆	70k	P ₆	70k
P ₇	220k	P ₇	220k
P ₈	300k	P ₈	300k

Now P₃ has completed the execⁿ, as well as P₆

(v) 120k available

(iv)

but this process P₉ won't be allocated because we don't have 100k contiguous memory, this is called external fragmentation.

Now, if a process P₉ comes with size of 100k (less than available free memory)

Now a process P₁₀ comes with size of 40k

divided from 50k mem. size

P ₁	
P ₂	
P ₁₀	→ 50k
P ₄	350k
P ₅	100k
P ₇	70k
P ₈	220k
	300k

To avoid the external fragmentation :-

1. Compaction: Moving all the processes towards the top or towards the bottom to make free available memory in a single contiguous block.
2. The compaction is undesirable to implement because it disturbs all the running processes in the memory.
3. Non-contiguous mem. management techniques.

Cosmos

classmate

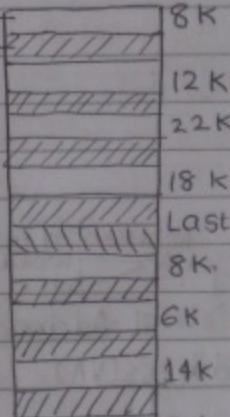
Date _____

Page _____

Partition Allocation Techniques:-

1. If more than one partition is freely available to accommodate the process, then the decision making will be done with the help of partition allocation techniques.
 - First Fit :- Allocate the process in a partition which is 1st sufficient from the top.
 - Best Fit :- Allocate the process in a partition which gives least internal fragmentation.
 - Worst Fit :- Allocate the process in a partition which gives most internal fragmentation.
 - Next Fit :- It also works like 1st fit, but it will search from last allocation point.

free
partition
busy
position



Memory

(i) P, comes with 16K
First Fit :- in 16K
Best Fit :- in 18K
Worst Fit :- in 36K
Next Fit :- in 36K

(Starting from
last allocation
pt., the First
fit for 16K i
36K)

Q. How many successive requests of 6K will be satisfied?

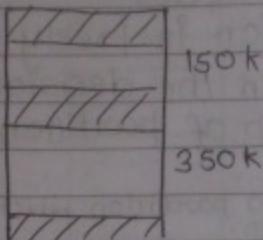
(First fit).

[using variable length
1 in 8K, 2 in 12K, 3 in 22K & so on. partition
scheme]

- (a) 17
(b) 18
 (c) 19
(d) 20

Q. Request from the processes are 300K, 25K, 125K, 50K resp.

Cosmos



The above request could be satisfied with? (Variable partition scheme)

Best fit:-

Ams.: First Fit but not Best Fit

Non-Contiguous :-

Paging :-

1. Logical Address Space (LAS) or Virtual Address Space (VA)
2. Logical Address (LA) or Virtual Address (VA)
3. Physical Address Space (PAS)
4. Physical Address (PA)

represented in the form
of words
bytes

represented in the form of bits.

represented in the form of words or bytes
represented in the form of bits.

e.g. • LA = 39 bits

LAS = 2^{39} words = 512 G words (if it is word addressable)

• LAS = 256 M words = $2^8 \times 2^{20}$ words = 2^{28} words.
∴ LA = 28 bits

• PA = 26 bits

PAS = 64 M words

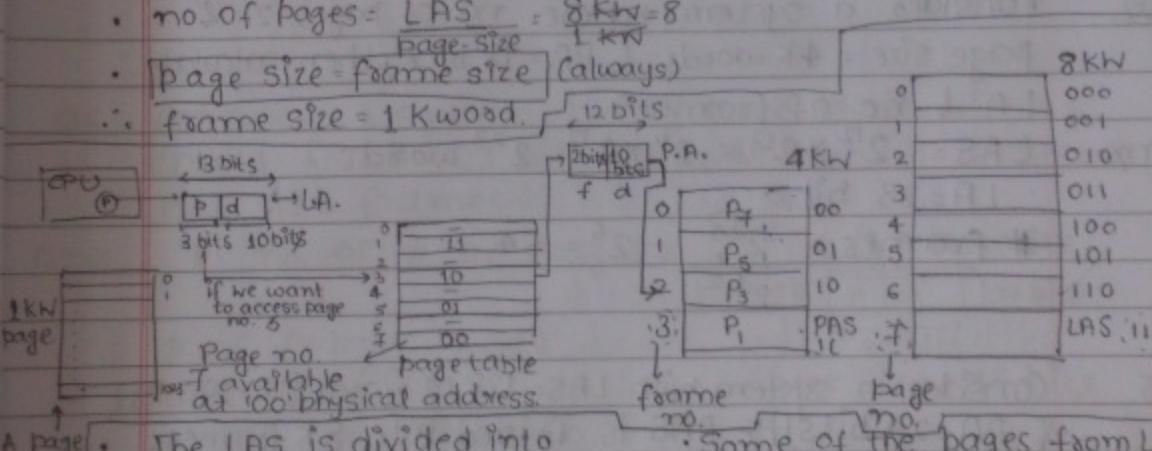
• PAS = 128 K words = 2^{17} words
∴ PA = 17 bits.

Cosmos

- ★ Actual process is available in Physical Address Space
- ★ CPU generates logical address.
- ★ CPU can't directly access the process with the logical address.
- ★ Mapping of logical address to physical address is called paging technique.

Let L.A. = 13 bits, LAS = 8K words & P.A. = 12 bits, PAS = 4K words } assuming word addressable memory

- page size = 1 K words (assume)
- no. of pages = $\frac{\text{LAS}}{\text{page-size}} = \frac{8\text{Kw}}{1\text{Kw}} = 8$
- page size = frame size (always)
- ∴ frame size = 1 K word.



- The LAS is divided into equal size pages.
- The PAS is divided into equal size frames.
- No. of entries in a page table = no. of pages in the LAS
- The page table entry contains the frame no.
- p = no. of bits req. to represent the pages of LAS or page no.
- d :- to represent particular words in a page or no. of bits req. to represent the page size of LAS or word no. of the page or page offset.
- f :- no. of bits req. to represent the frames of PAS or frame no.
- so, f will tell the frame no. & d is the frame offset in that particular frame.

Cosmos

P 4 Q1. Consider a system with LA= 27 bits & Physical Address = 21 bits & page size = 4K word, calculate no. of pages & no. of frames.

Ans. # pages = $\frac{2^{27} \text{ words}}{2^{12} \text{ words (in 1 page)}} = 2^{15} = 32 \text{ K}$

frames = $\frac{2^{21} \text{ words}}{2^{12} \text{ words (in 1 frame)}} = 2^9 = 512$

Q2. Consider a system where no. of pages = 2k & page size = 4K words & PA = 18 bits, then calculate LA & no. of frames?

Ans. LAS = $2^{12} \times 2^{13} / 2^{11} \times 2^{12} = 2^{23} \text{ words}$

∴ LA = 23 bits

frames = $\frac{2^{18}}{2^{12}} = 2^6 = 64$

Q3. Consider a system with LAS = 128 M words [LA = 27 bits & PA = 24 bits], the PAS is divided into 8K frames. What is the page size & how many pages in the LAS?

Ans. page size = $\frac{8 \times 2^{10}}{2^{13}} = 2^{13}$, ^{for PAS}
(as frame size) $\frac{2^{24}}{2^{13}} = 2^{11} = 2 \text{ K words}$

pages = $\frac{2^{27}}{2^{11}} = 2^{16} = 64 \text{ K}$ ^{no. of frames}

Q4. Consider a system with LA = 32 bits & PAS = 64 MB (PA = 2²⁶), page size = 4 KB, the mem. is byte addressable, the page table entry size is 2 bytes, what is the approx. page table size in bytes?

- (a) 1 MB (c) 4 MB
(b) 2 MB (d) 8 MB

Cosmos

classmate

Date _____

Page _____

Ans. Page table entry size = 16 bits

$$\therefore \# \text{ of pages} = 2^{36}$$

$$LA = 32 \text{ bits}$$

$$LAS = 2^{32} \text{ Bytes}$$

$$\# \text{ of frames} =$$

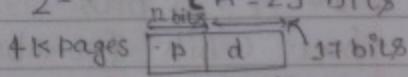
$$\# \text{ of pages} = \frac{2^{32}}{2^{12}} = 2^{20} = 1 \text{ MB}$$

$$\therefore \text{page table size} = 2^8 \times 1 \text{ MB} = 2 \text{ MB}$$

No. of occupied page table entries = No. of frames (max.)

Q5. Consider a system having page table with 4K entries & LA = 29 bits, what is the PA if system has 512 frames?

Ans. ~~# of pages~~ = 2^{12} $LA = 29 \text{ bits}$

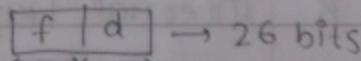


$$\# \text{ of frames} = 512$$

$$PA = \cancel{PAS} \cdot \cancel{\# \text{ of frames}}$$

$$\therefore PA = 26 \text{ bits}$$

$$\therefore \text{page size} = 17 \text{ bits}$$

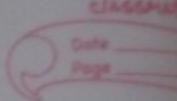


(from 512 frames)

Q6. Consider a system with LAS = 256 MB [LA = ~~2²⁸~~ bits] & PA = 22 bits, the PAS is divided into 8 KB frames (page size). The mem. is byte addressable, what is the # of pages?

Ans. $\# \text{ of pages} = \frac{2^{28}}{2^{13}} = 2^{15} = 32 \text{ K}$

Cosmos

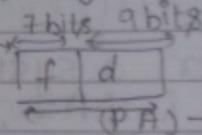


- Q. Consider a system with LAS = PAS = 2^{16} Bytes. The page size = 512 Bytes. The page table entry size is 2 Bytes. The mem. is byte addressable. The page table entry contains besides other info:
- 1 bit \rightarrow valid/invalid
 - 1 bit \rightarrow reference
 - 3 bits \rightarrow page protection
 - 1 bit \rightarrow dirty bit

How many bits are still available in the page table entry to store ageing info.

Ans. PA = LA = 16 bits.

$$\# \text{ of frames} = \frac{2^{16}}{2^9} = 2^7$$



. 7 bits are req. for frame no.

$$16 - f = 9 \text{ bits}$$

but 6 bits for above info.

$$\therefore \text{bits for ageing} = 9 - 6 = 3 \text{ bits.}$$



Internal Fragmentation = $\frac{P}{2}$

P: page size

- Q. Consider a system with LAS = PAS = 5 bytes. The page size = P bytes, page table entry size = e bytes. What is the optimal value of page size by minimizing the memory overhead of page table size and internal fragmentation in the paging.

- (a) $P = \sqrt{2Se^2}$
- (b) $P = \sqrt{2(Se)^2}$
- (c) $P = \sqrt{2Se}$
- (d) $P = \sqrt{2S^2e}$

* All the page table of the processes are placed in the main memory.

classmate _____

Date _____

Page _____

Cosmos

$$\text{Ans } LAS = PAS = S \text{ bytes}$$

of pages F

$$\# \text{ of frames} = \frac{S}{b} = \# \text{ of pages}$$

$$\text{page table entry size} = e \text{ bytes}$$

$$\text{So, the total size of page table} = \frac{S}{b} \cdot e \text{ bytes}$$

$$\text{memory overhead} = \text{page table size} + \text{Internal frag.}$$

$$= \frac{S \cdot e}{b} + \frac{P}{2}$$

$$\frac{d}{dp} \left(\frac{S \cdot e}{b} + \frac{P}{2} \right) = 0$$

$$Se \log b + \frac{1}{2} = 0 \quad Se \left(-\frac{1}{b^2} \right) + \frac{1}{2} = 0$$

$$Se \log b = -\frac{1}{2} \quad 2Se = \frac{1}{b^2} \quad \Rightarrow b = \sqrt{2Se}$$

Q. * Main mem. access time = 'm'

[If page tables are kept in main mem.

$$\text{Effective memory Access time} = m + m = 2m$$

\uparrow to access page \uparrow to access the page
table

If TLB is added to improve the performance

The TLB contains the frequently referred page nos. & corresponding frame nos., the TLB is implemented with the help of associative register

$$\text{TLB access time} = 'c'$$

$$\text{TLB hit ration} = 'x'$$

$$\therefore \text{Effective mem. access time} =$$

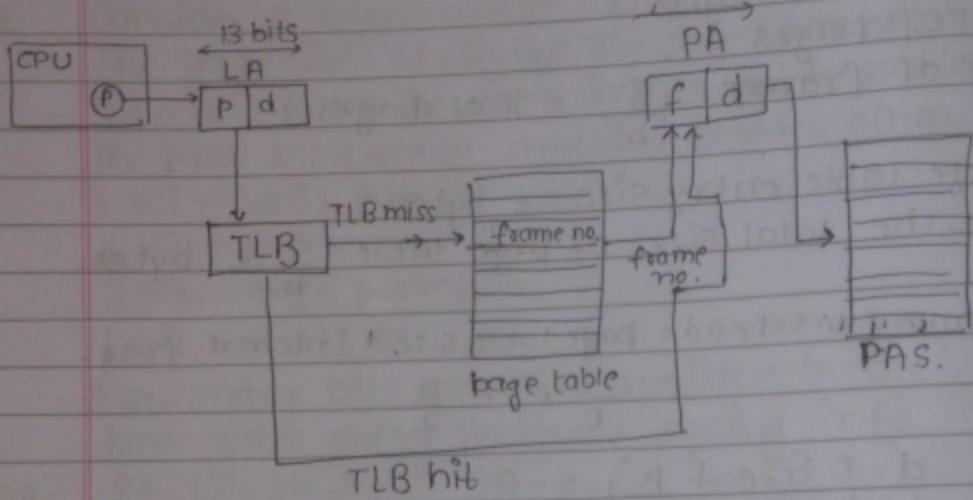
$$[x(c+m)] + [(1-x)(c+m+m)]$$

↓ when hit ↓ when miss ↓ PAS access ↓ page table access

Cosmos

classmate

Date _____
Page _____



Q. Consider a system with main mem. access time = 100 ns & TLB access time = 20 ns & TLB hit ratio = 95%, what is effec. mem. access time with & w/o TLB.

Ans. W/o TLB = $2 \times m = 200 \text{ ns}$

$$\begin{aligned} \text{with TLB} &= 0.95 \times (20 + 100) + \\ &\quad 0.05 \times (20 + 100 + 100) \\ &= 0.95 \times 120 + 0.05 \times 220 \\ &= 114 + 11 \\ &= 125 \text{ ns.} \end{aligned}$$

Q. What hit ratio is req. to reduce effec. mem. access time from 300 ns (w/o TLB) to 250 ns (with TLB), the TLB access time = 60 ns.

Ans. $2m = 300 \text{ ns}$

$m = 150 \text{ ns}$

$$250 = x \cdot (210) + (1-x) [360]$$

$$250 = 210x + 360 - 360x$$

$$110 = 150x$$

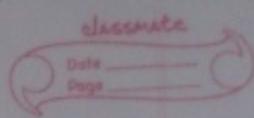
$$x = 11/15 = 0.73 \approx 0.73$$

$$\begin{array}{r} 73 \\ \times 15 \\ \hline 365 \\ -210 \\ \hline 110 \end{array}$$

$$\begin{array}{r} 110 \\ -105 \\ \hline 50 \end{array}$$

$$\begin{array}{r} 50 \\ \times 15 \\ \hline 750 \end{array}$$

★ Every process has its own page table.
To avoid the overhead of maintaining the large page tables, the multilevel paging will be implemented.



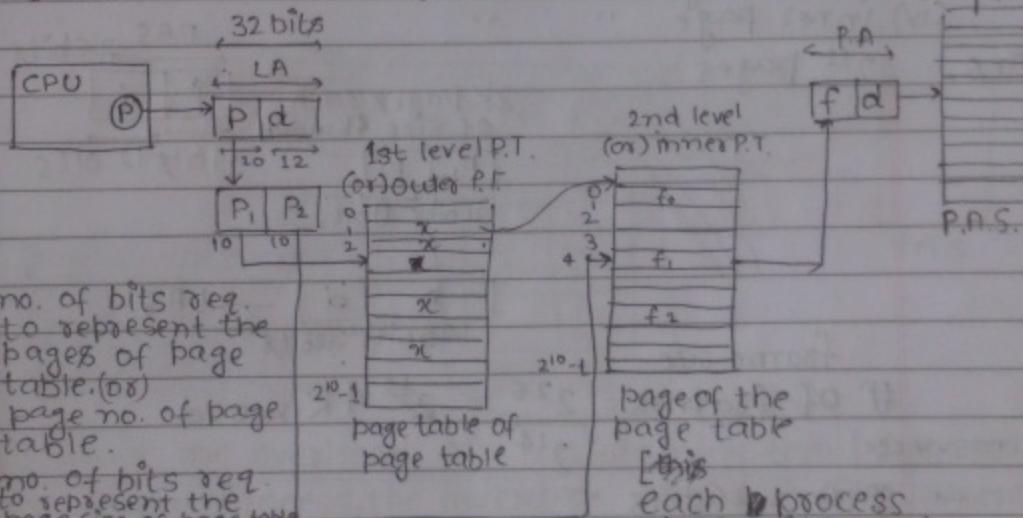
Cosmos

Multilevel Paging

Q. Consider a system with logical address = 32 bits & page size = 4 K words. The page table entry size = 4 B, what is the page table size?

Ans. # of pages = $\frac{2^{32}}{2^{12}} = 2^{20} = 1 \text{ M}$

$1 \text{ M} \times 4 \text{ B} = 4 \text{ MB}$:- page table size



p₁: no. of bits req. to represent the pages of page table. (10)

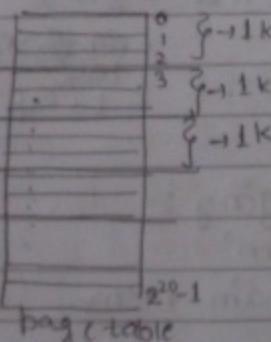
page no. of page table.

p₂: no. of bits req. to represent the page size of page table.

x: - address of page of page table
page size of page table = 1 KW

page of the page table [this each process has different inner page table.]

single level paging :-



We are dividing the page table with page size of 1 K

1K → 1K : # of pages in page

$$\text{table} = \frac{2^{20}}{2^{10}} = 2^{10} \text{ pages in } 2^{10} \text{ page table.}$$

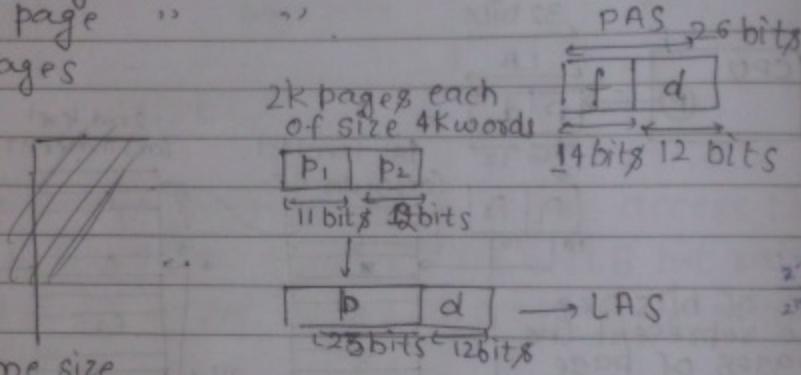
Cosmos

Date _____
Page _____

Q. Consider a system with 2 level paging applicable
the page table is divided into 2K pages each of
size 4K words, if PAS is 64M words which is
divided into 16K frames. The page table entry
size is 4B, then calculate

- (i) length of L.A.
- (ii) length of P.A.
- (iii) outer page table size
- (iv) inner page

Ans. 2^{11} pages



$$\# \text{ of frames} = \frac{2^{26}}{2^{14}} = 2^2 \text{ or } 4 \text{ K word}$$

My answers:-

$$\rightarrow \text{P.A.} = 26 \text{ bits} \checkmark$$

$$\rightarrow \text{L.A.} = 35 \text{ bits}$$

$$\rightarrow \text{outer page table size} = 2^{12} \times 2^{11} = 2^{23}$$

$$\rightarrow \text{inner page table size} = 2^{12} \times 2^{11} = 2^{23}$$

$$= 8,000,000 \text{ bytes}$$

$$= 16 \text{ KB}$$

★ Performance of 2-level paging :-

Main mem. access time = 'm'

If the P.T.'s are kept in main mem.

Effec. main mem. = 3M

Cosmos

descrete

Data
Page

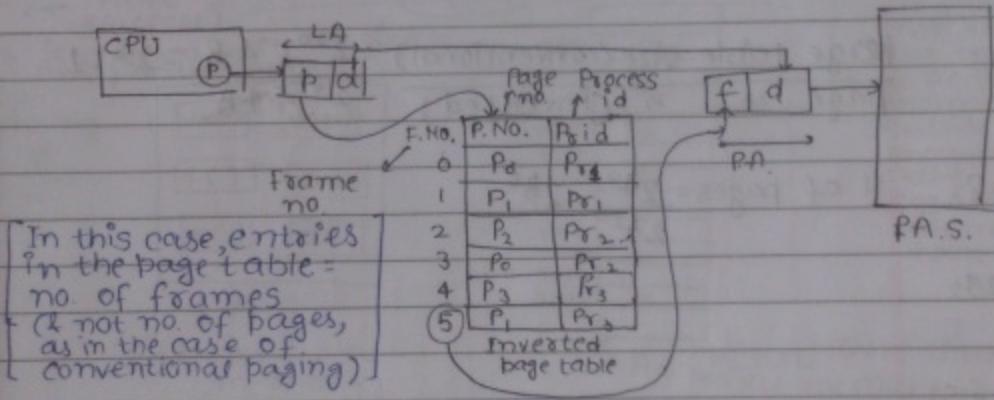
If TLB is added,
the performance :-

TLB access time = 'c'

TLB hit ratio = 'x'

$$\therefore E_{MNT} = x(c+m) + (1-x)(c+3m)$$

Inverted Paging



- ★ To avoid the overhead of maintaining the page table for every process, the inverted paging is implemented. In the inverted paging, only 1 page table is maintained for all the processes.
- ★ No. of entries in the page table is same as the no. of frames in the P.A.S.

- Q Consider a system with LA = 34 bits & PA = 29 bits & page size is 16 KB. The page table entry size = 8B. Calculate the page table size in conventional paging & inverted paging!

Ans. # of pages = $\frac{2^{34}}{2^{14}} = 2^{20}$ Page table size = $8B \times 1M = 8MB$
(in conventional)

of frames = $\frac{2^{29}}{2^{14}} = 2^{15}$ Page table size = $32KB \times 8B = 256B$
(in inverted)

Cosmos

classmate

Date _____
Page _____

Page 53
Q 31

$$L.A. = 32 \text{ bits}$$

$$\text{Page size} = 4 \text{ KB}$$

$$P.A. = 128 \text{ KB}$$

conventional page table:-

$$\# \text{ of pages} = \frac{2^{32}}{2^{12}} = 2^{20}$$

inverted

$$\# \text{ of frames} = \frac{2^{17}}{2^{12}} = 2^5$$

$$\begin{aligned} \text{page table size (conventional)} &= \frac{2^{20} \times 4B}{2^5 \times 4B} = 2^{15}:1 \\ \text{page } \gg \text{ } \gg \text{ (inverted)} & \end{aligned}$$

Q28. # of pages = $\frac{2^{48}}{2^{12}} = 2^{36}$

Q29.

① page 51
Q5.

$$\text{Page size} = 4 \text{ KB}$$

$$\# \text{ of pages} = 2^{20}$$

$$LA = 32 \text{ bit}$$

$$PA = 30 \text{ bit}$$

$$\# \text{ of frames} = \frac{2^{30}}{2^{12}} = 2^{18}$$

$$\begin{aligned} \therefore \text{page table size} &= 2^{18} \times (18+12) \text{ bits} && \text{↑ bits for no. of frames} \\ &= 2^{18} \times (30) \text{ bits} && (= \text{no. of pages in inverted page table}) \\ &= 2^{23} \text{ bits} && \approx 2^5 \\ &= 2^{20} \text{ B} && \end{aligned}$$

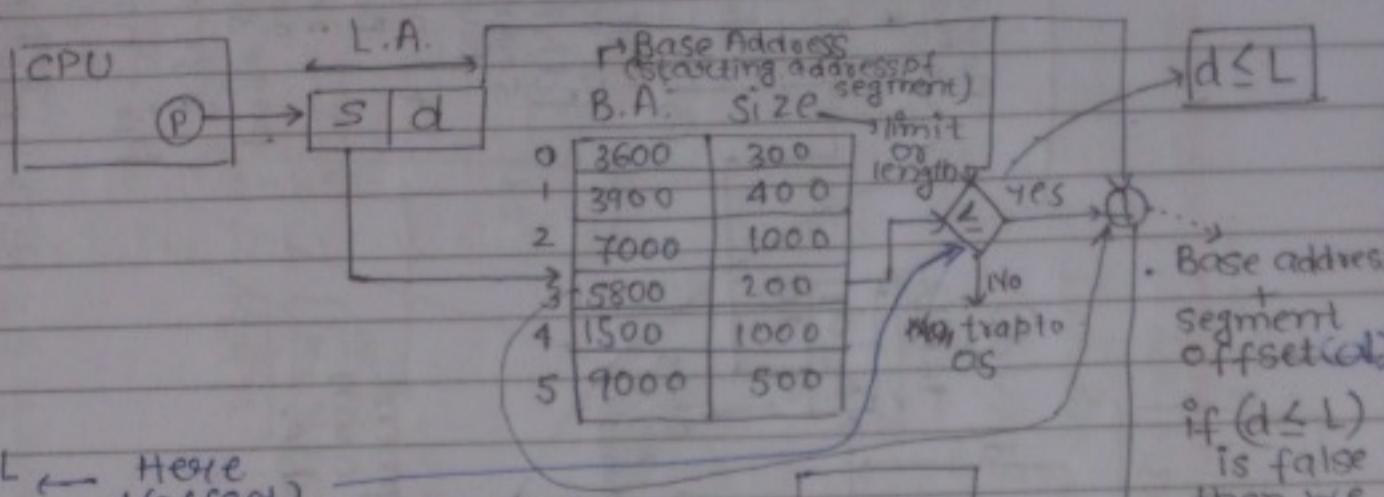
Cosmos

classmate

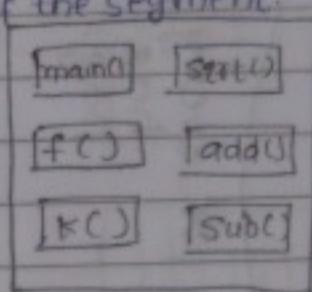
Date _____

Page _____

Segmentation



if $d \leq L$ ← Here
 then access is checked with the size (or
 valid limit) of the segment.
 otherwise it is accessing something
 outside the segment.



L.A.S.

(User's view of memory)

1500	S4
2000	
3600	S0
3900	S1
4300	
5000	S3
6000	
7000	S2
8000	
9000	S5
9500	

P.A.S (Actual one dimensional memory)

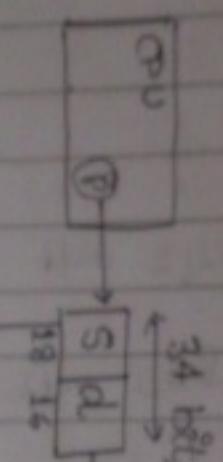
- ★ Paging doesn't follow user's view of memory allocation. To achieve the user's view of memory allocation, the segmentation will be implemented. The L.A.S. is divided into various segments, the segments will vary in the size.

s:- no. of bits required to represent the segments of L.A.S.

d:- no. of bits req. to represent the size of segment or segment offset.

Degraded Paging

paging on segment when segment size increases.

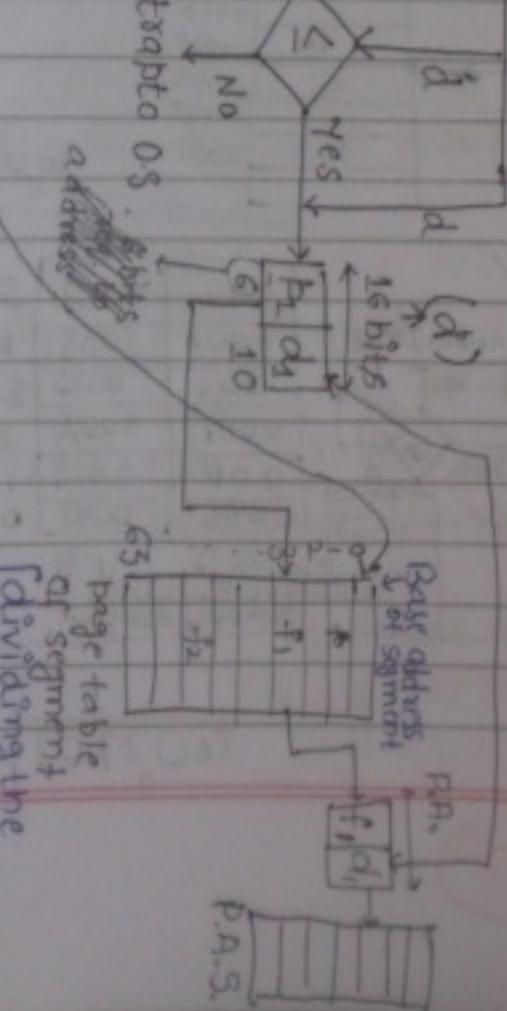


LA = 34 bits
 $\langle s, d \rangle = \langle 18, 16 \rangle$

segment size = 2^{16} bytes
 $= 64 \text{ KB}$

64KB

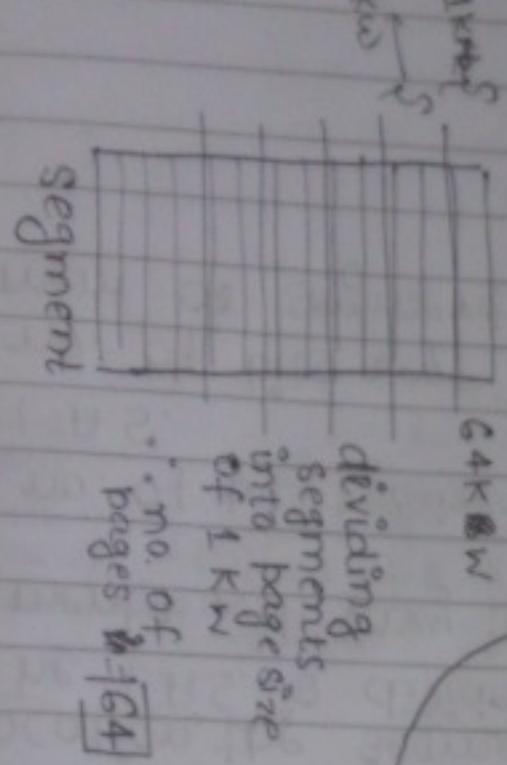
segment table



[dividing the segment into pages]

* address of page table of respective segment

* no. of bits to delineate



* every segment has its own page table.

segment

dividing segments into page size of 1 KB

* no. of pages = 164

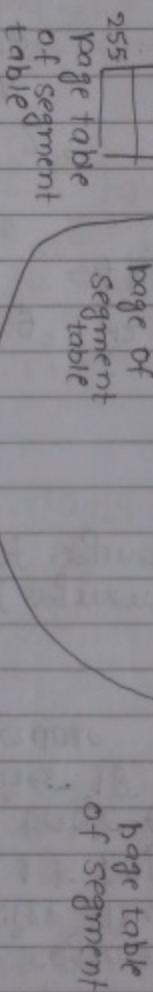
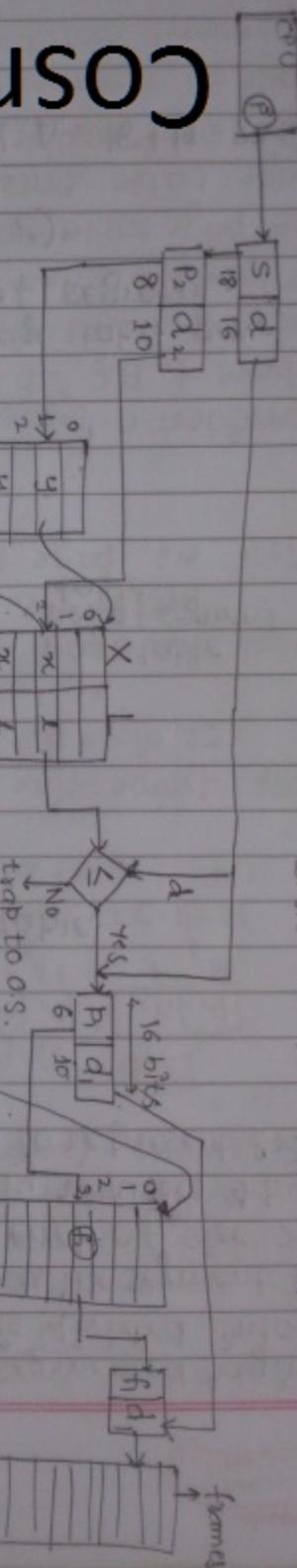
COSMOS

COSMOS

y : address of page of segmented table
 size of segment table = 1 KB

Paging on Segment basis

paging of segment when segment size increases & also paging on segment to handle when S.T. size increases



page of segment table

page size of segment table -
 1 KB

no. of pages

of segment +

table - $2^{18} = 2^8 = 256$

$$2^{18-1}$$

x : address of page table of respective page size of segment - 1 KB

P_2 : no. of bits req. to represent the pages of segment table
 or page no. of segment table.

d_1 : no. of bits req. to represent the page size of segment table.

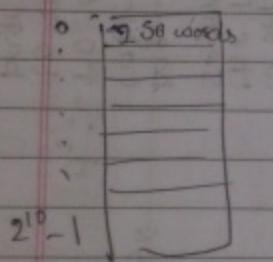
classmate

classmate

Cosmos

Q. Consider a system using segmented paging architecture. The segment is divided into 1K pages each of size 512 words. The segment table is divided into 1 K pages each of size 256 words. The page table entry requires 4 bytes, & frame no. requires 18 bits to represent frames of P.A.S., then calculate

- length of L.A.
- length of P.A.
- page table size of segment
- page table size of segment table.

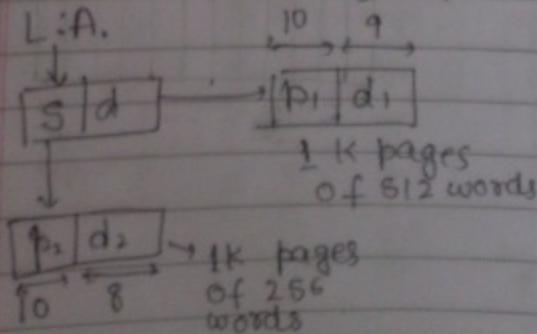


$$\begin{aligned} 2^{10} \times 2^2 &= 2^{12} \text{ B} \\ &= 4 \text{ KB} \rightarrow \text{page table size of segment table} \end{aligned}$$

$$\begin{aligned} \# \text{ of frames} &= 2^{18} \\ 18 + 9 &= 27 \text{ bits} \\ \therefore \text{P.A.} &= 27 \text{ bits.} \end{aligned}$$

$$2^{10} \times 2^2 = 2^{12} \text{ B} = 4 \text{ KB} \rightarrow \text{page table size of segment.}$$

↓
(as page table entry requires 4B = 2²)



Cosmos

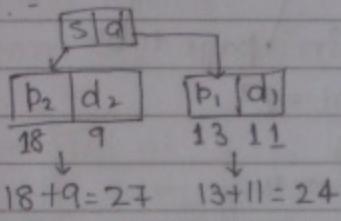
classmate

Date _____

Page _____

Q. Consider a segmented paging arch. Where segment is divided into 8k pages each of size 2K words. The segment table is divided into 256 K pages each of size 512 words. The page table entry requires 64 bits of size & frame no. requires 22 bits to represent frames of P.A.S, then calculate.

Ans. • L.A. =



$$\therefore \text{L.A.} = 27 + 24 = 51 \text{ bits}$$

• P.T.E. frame size =

$$22 \text{ bits} + d_1 \text{ bits} = 22 + 11 = 33 \text{ bits}$$

now, size of page table of segment table =

$$2^3 \text{ B} \times 2^{18} = 2^{21} \text{ B} = 2 \text{ MB}$$

size of page table of segment = $2^3 \text{ B} \times 2^{13}$
 $= 2^{16} \text{ B} = 64 \text{ KB}$

Q.

Consider a system with segmented paging arch. Where L.A.S. = P.A.S. = 2^{16} bytes, the L.A.S. is divided into 8 equal size segments, the segment is divided into equal size pages which are of powers of 2. The page tables are stored in the main mem. A page table entry requires 2B. The memory is byte addressable. What must be the size of the page of segment in bytes, so that page table of segment exactly fits in 1 page frame.

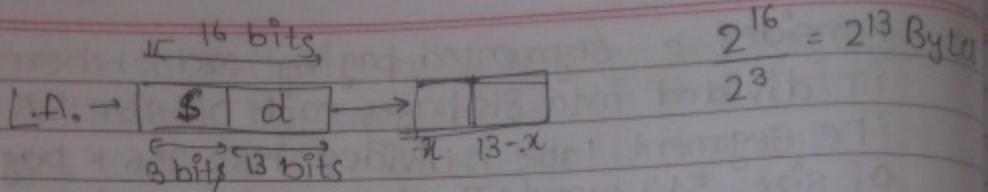
Ans. frame size = $\frac{2^{16}}{2^3} = 1024 \text{ bytes}$

Cosmos

classmate _____

Date _____

Page _____



1 segment is divided in page size which is powers of 2

$$\therefore 2^{13} \text{ B} = n \times 2^B$$

$n = 2^{12}$ pages in page table

Let page size of segment = 2^x

$$\text{frame size} = \frac{2^{16} \text{ Bytes}}{\text{no. of frames}}$$

page size of segment = frame size of P.A.S.

$$\therefore 2^{16} = 2^x$$

page table size of segment = page size of segment

$$\left(\frac{2^{13}}{2^x}\right) \times 2^B = 2^B$$

$$= 2^x$$

$$\left(\frac{2^{13}}{2^x}\right) \times 2^B = 2^x$$

each entry
of 2^B
each.

no. of
entries
in page
table

$$\therefore x = F$$

$$2^7 = 128 \text{ B}$$

Date _____

Cosmos

classmate

Date _____
Page _____

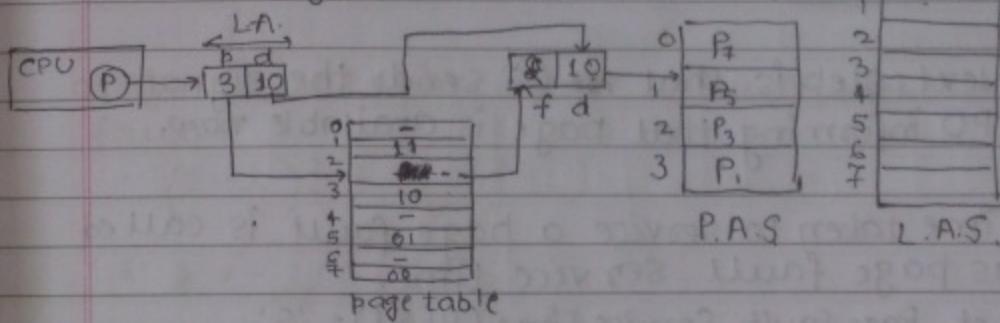
01.07.12

Virtual Memory

Defn: Virtual Memory gives an illusion to the program that programs of larger size than actual physical memory can be executed.

It is implemented using:-

- Demand Paging
- Demand Segmentation



★ If we allocate all the pages of one process into physical mem., then less no. of processes can be allocated to RAM, hence decreasing the degree of multiprogramming.

Page fault: The processor is trying to access the page which is not currently available in the main memory.

- When page fault occurs, then signal is sent to signalling processor requires a page currently present in the phy.P.A.S.
- Now, OS validates the page request of the process.
- Then, OS searches for the required page in L.A.S.
- When OS found the req. page, it brings the page from L.A.S. to P.A.S.
- Let suppose, processor req. P₄ page not currently in P.A.S., brings P₄ in place of P₃ (assumption) & changes the P.A.S. [We use page replacement algorithm for this]
- Now req. page is in P.A.S., we now update the page table.

Cosmos

classmate

Date _____

Page _____

table.

New page table:

0	-
1	11
2	-
3	10 -
4	010
5	01
6	-
7	00

- Next step is that OS sends the signal to CPU informing that page is available now.

- * Time taken to service a page fault is called as page fault service time.

Let Page Fault Service Time (PFST) = 'S'

& Main mem. access Time (MMAT) = 'M'

& Page Fault Rate = 'P'

Effective memory access time =

$$(1-P)(2M) + P(M+T)$$

$$(1-P) \times M + P \times S$$

Only one

M & not

2M, because

we ignore
the page table
access time.

actually
 $(S+3M)$

$$M + M + M$$

page table access time
main mem. access time
page table access time
(2nd time)

When there access time
was page fault time hit ratio

- Q. Consider a system with page fault rate of 99% & PFST is 10 msec. & main mem. access time is 1 msec, what is effec. mem. access time.

$$\text{EMAT} = 0.99 \times 1 + (0.01) \times 10$$

$$= 0.99 + 0.1 = 1.09 \text{ msec.}$$

Cosmos

classmate

Date _____

Page _____

Q. Suppose if an instr takes i usec & additional j usec. If page fault occurs. If a page fault occurs on an avg. for every k instr, then what is the effec. instr access time?

Ans.

$$\begin{aligned} & \left(1 - \frac{k}{n}\right)i + \frac{k}{n}(j+i) \\ &= 1 - \cancel{\frac{k}{n}i} + \cancel{\frac{k}{n}j} + \cancel{\frac{k}{n}i} \\ &= 1 + \end{aligned}$$

Page fault \rightarrow for every k instr, there is a page fault.

$$\begin{aligned} & \therefore \left(1 - \frac{1}{k}\right)i + \frac{1}{k}(j+i) \\ &= \cancel{i} - \cancel{\frac{i}{k}} + \cancel{j} + \cancel{\frac{i}{k}} \\ &= i + \frac{j}{k} \end{aligned}$$

Q. Let the page fault service time be 10 msec & mem. access time is 20 ns. If 1 page fault is generated for every 10^6 mem. accesses, what is the effec. access time of mem.

Ans.

$$\begin{aligned} & (1 - 10^{-6}) \times 20 \text{ ns} + 10^{-6} \times 10 \times 10^6 \\ &= 20 \text{ ns} + 10 \text{ ns} \\ &= 30 \text{ ns} \end{aligned}$$

Cosmos

classmate

Date _____

Page _____

- Q. Consider a demand page env. where it takes 8ms to service a page fault, if either an empty frame is available or replaced page is not to be modified & it takes 20 msec. if replaced page is modified. Assuming main mem. access time = 1 ms. & further assume the page to be replaced is modified 70% of the time, what is the max. acceptable page fault rate to get effec. mem. access time not more than 2 msec.

Ans. $0.7 \times p \times 20 + 0.3 \times p \times 8 + (1-p) \times 1 < 2$
 $14.7p + 2.7p + 1 - p < 2$

$\frac{17.4p}{21.4} < 1$

$p < \frac{1}{15.4}$

$p < 0.06493$

(when main
mem. access
time is neglected
when page fault
occurs)

or

$$0.7 \times p \times 21 + 0.3 \times p \times 9 + (1-p) \times 1 < 2$$

$$14.7p + 2.7p + 1 - p < 2$$

$$16.4p < 1$$

$$p < 0.06097$$

(when main
mem. access
time is not
neglected
when page fault
occurs).

★ Page modified means:-

when page is modified in P.A.S., & not in L.A.S., then the page is first saved in L.A.S. & then replaced.

Cosmos

classmate

Date _____

Page _____

- Q. Consider a system where page fault service time = 200 msec. & main mem. access time = 10 msec., the TLB is added to improve the performance & 80% references are available in TLB & that of remaining 10% causes page faults, the TLB access time is negligible, what is the effec. memory. access time?

→ TLB hit → TLB Miss

Ans.

$$\begin{aligned} & 0.8 \times 10 + 0.2 \times (10(0.9) + 0.1 \times 200) \\ &= 8 + 0.2(9 + 20) \\ &= 8 + 0.2(29) \\ &= 8 + 5.8 \\ &= 13.8 \text{ ms} \end{aligned}$$

out of the
20%, TLB Miss,
10% is causing
page fault.

$$\begin{aligned} & 0.8 \times 10 + 0.2 (0.1 \times 10 + 0.1 \times 200) \\ &= 8 + 0.2 (1 + 20) \\ &= 8 + 4.2 \\ &= 12.2 \end{aligned}$$

Page Replacement Algorithm :-

Reference String :-

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

↳ page number
referenced by
the process

Assumption :- The frames allocated to the process = 4.

*:- page fault occurs
when it

classmate

Cosmos

FIFO (First In First Out)

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
			2	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1
			1	1	1	1	1	1	1	X	0	0	0	0	0	0	0	0	0
			0	0	0	0	0	4	4	4	4	4	4	4	4	4	4	7	7
			7	7	7	7	X	3	3	3	3	3	3	3	3	3	2	2	2
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

now # of frames = 3

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
			1	1	1	Y	0	0	0	3	3	3	3	3	2	2	2	2	1
			0	0	0	0	3	3	3	8	2	2	2	2	2	1	1	1	X
			7	7	X	2	2	2	2	X	4	4	4	4	X	0	0	0	0
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

Now, reference string:-

1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

of frames = 3

1	2	3	4	1	2	5	1	2	3	4	5
			3	3	3	2	2	2	2	2	4
			2	2	2	1	1	1	1	X	3
			1	1	X	4	4	4	5	5	5
*	*	*	*	*	*	*	*	*	*	*	*

of frames = 4

1	2	3	4	1	2	5	1	2	3	4	5
			4	4	4	4	4	X	3	3	3
			3	3	3	3	3	8	2	2	2
			2	2	2	2	2	X	1	1	X
			1	1	1	1	1	Y	5	5	5
*	*	*	*	*	*	*	*	*	8	4	4

Cosmos

classmate

Date _____
Page _____

↳ Belady's Anomaly:-

If the no. of frames allocated to the process increases the page fault increases.

Optimal Page Replacement:-

In the event of page fault, replace the page which is not used for longest duration of time in future.

of frames = 4.

	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
4th		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3rd		1	1	1	2	4	4	4	4	4	4	4	4	4	4	4	4	7	7	7
2nd		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1st	7	7	7	7	3	3	3	3	3	3	3	3	3	3	3	3	1	1	1	1
↑ position	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

7 is used
not used
for longest
duration of
time in future
from 7, 0, 1, 2

here
both
3 & 4

not used
in future,
now from
3 & 4, 3 comes
1st in FIFO because
last replaced is from
3rd position, . 3 → at 1st pos
& 4 → at 3rd pos, we have
to start from 4th pos. (1 plus from
last replaced pos.)

Cosmos

Least Recently Used:

In the event of page-fault, replace the page which is ~~not~~ least recently used.

7	0	1	2	0
		2	2	
	1		1	
	0	0	0	
7	7	7	7	7

of frames

= 3

Page 59

Q29.

030.

1	2	3	4	5	6	9	1
	X	3	3	8	5	5	1
2	2	2	7	7	7	6	6
1	1	1	1	4	4	3	3

1 2 1 3 7 4 5 6 3 1

Cosmos

classmate

Date _____

Page _____

page
58
Date
1/1/2014

Most Recently Used (MRU) :-

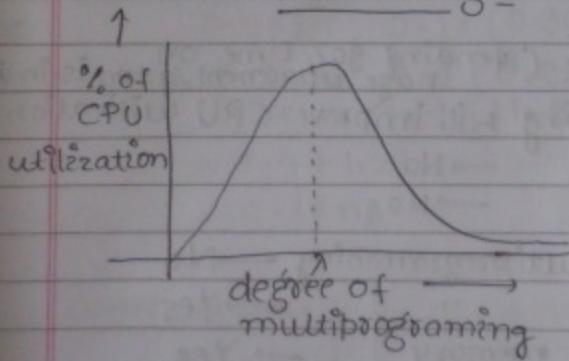
frames = 4

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	0	0	0
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

frames = 3

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	0
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

Thrashing -



Let free frames = 300

& no. of processes = 100,

then every process gets 3 frames each (with max. deg. of multiprogramm)

but if we have no. of processes = 300, then every process gets only 1 frame & page faults occur more frequently & most of the time is spent

Cosmos

classmate

Date _____

Page _____

- ① Initially when the degree of multiprogramming ≤ 2 , the CPU utilization increases.
- ② If degree of multiprogramming is drastically increased after 2, then CPU utilization goes on decreasing.
- ③ This situation is called Thrashing.
- ④ In thrashing, the system will spend more time only on the page replacement.

Causes Of Thrashing:-

- ① High degree of multiprogramming.
- ② Lack of frames.

Q. Consider a system with the below statistics
Observed,

→ CPU utilization $\rightarrow 10\%$.

→ paging on disk $\rightarrow 90\%$. (Spending 90% time on page replacement, system is in thrash)
Which of the following will improve CPU utilization.

- (a) Install faster CPU \rightarrow No
- (b) Install bigger Disk \rightarrow No
- (c) Increase degree of multiprogramming \rightarrow No
- (d) Decrease " " " \rightarrow Yes
- (e) Install more main memory \rightarrow Yes
- (f) Decrease page size \rightarrow No (Page fault because no. of frames increases),
Allocated to the pages are constant so decrease in page size causes the page faults to memory
- (g) Increase page size \rightarrow Yes likely

Cosmos

classmate

Date _____

Page _____

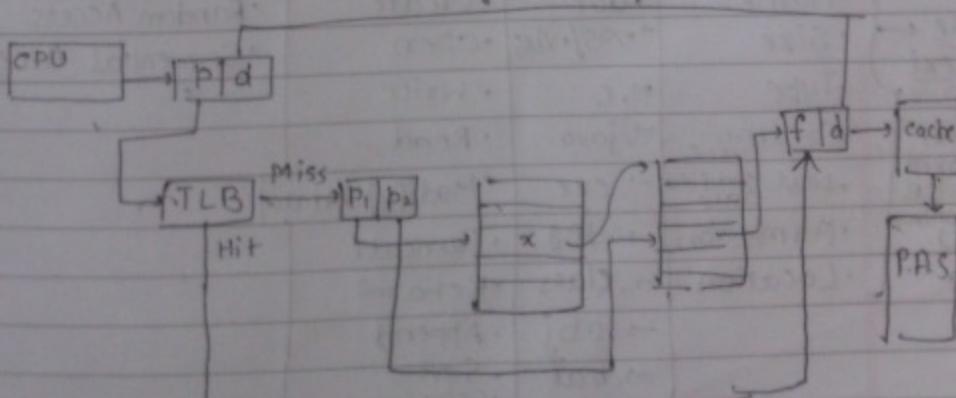
Q. The processor uses two-level page table for Virtual to Physical Address Translation. Page tables of both the levels are stored in the main mem. L.A = P.A. = 32 bits. The memory is byte-addressable. For Virtual to Physical Address translation the 10 most significant bits of V.A. are used as index into 1st level page table, while the next 10 bits are used as index into 2nd level page table. The 12 least significant bits of V.A. are used as offset within the page. Assume that page table entries in both the levels of page table are 4B wide. Further, the processor has TLB with hit ratio of 96%. The TLB caches recently used Virtual page nos. & corresponding Physical page nos. The processor also has a physically addressed cache with hit ratio of 90%, the main mem. access time = 10ns, cache access time = 1ns, TLB access time = 1ns. What is effec. mem. access time?

Ans. My answer:-

$$\begin{aligned} & 0.9 \times 0.96 \times 10 + 0.1 \times (0.96 \times 10 + 0.04 \times (20 + 10)) \\ &= 0.9 + 0.1(0.96 \times 10 + 0.04 \times 32) \\ &= 0.9 + 0.1(11.52 + 1.28) \\ &= 0.9 + 1.28 \\ &= 2.18 \text{ ns} \end{aligned}$$

→ wrong
(correct
only when Cache
is logically addressed)

Correct Answer:-



Cosmos

classmate

Date _____

Page _____

$$0.96(1+10) + 0.04(0.9x)$$

$$0.96(1 + 0.9 \times 1) + 0.1 \times (1+10) +$$

$$0.04(1 + 10 + 10 + 0.9 \times 1) + 0.1 \times (1+10)$$

$$= 0.96(1 + 0.9 + 1.1) +$$

$$0.04(1 + 20 + 0.9 + 1.1)$$

$$= 0.96(3) + 0.04(23)$$

$$= 2.88 + 0.92$$

$$= 3.8 \text{ ms}$$

Page
no.

39 Q40.

File & Device Management



→ Hard Disk
→ Tape

File :- Collection of logically related entities.

File Context	Attributes:-	Types:-	Operations:-	Access Methods :-
	• Name	→ .doc	• Create	• Random Access
	• Size	→ .xls/.xlsx	• Open	• Sequential Access
	• Type	→ .c	• Write	
	• Creation Date	→ .java	• Read	
	• Last Modified Date	→ .exe	• Modify/Update	
	• Permission	→ .jpg	• Truncate	
	• Location	→ .class	• Rename	
		→ .obj	• Append	
		→ .bat	• Save	
			• Copy	
			• Close	

Cosmos

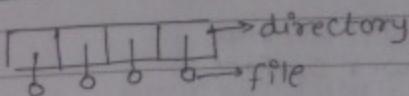
classmate

Date _____

Page _____

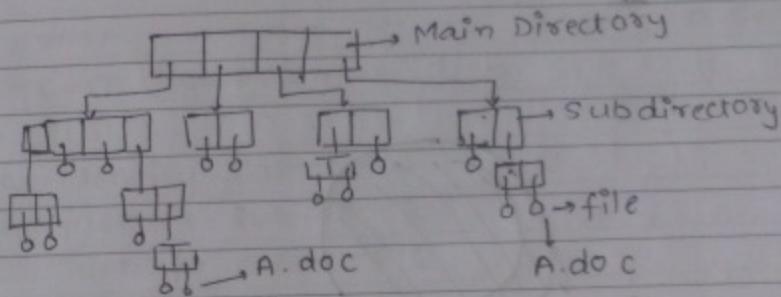
Directory Structure :-

(i) Single Level Directory :-



- We can't have two files with same name.
- Searching time of file will increase.

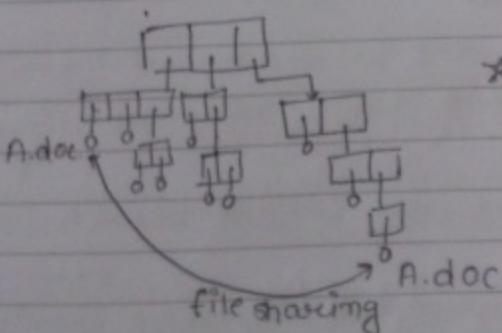
(ii) Tree Level Directory :-



- Easy Classification
- Less Search Time
- If we change something in A.doc at one place, we explicitly need to perform same changes at other place.

(iii) Acyclic directory Structure :-

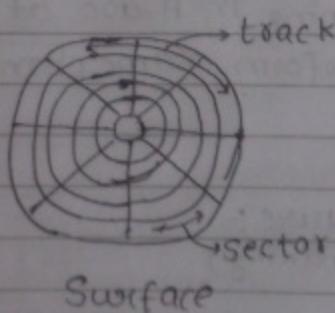
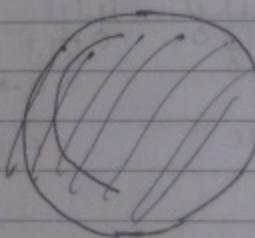
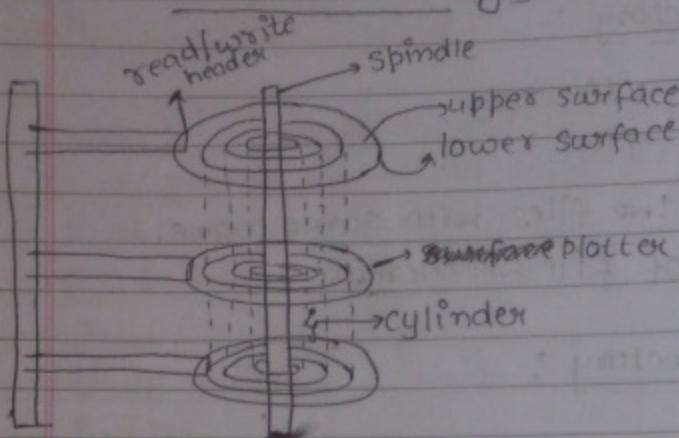
(allows the file sharing)



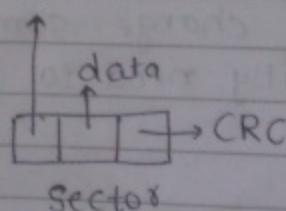
* In this structure, if we changes A.doc at one place, then it will be modified same at other place itself.

Cosmos

Disk Scheduling



header or pre-amble



Cosmos

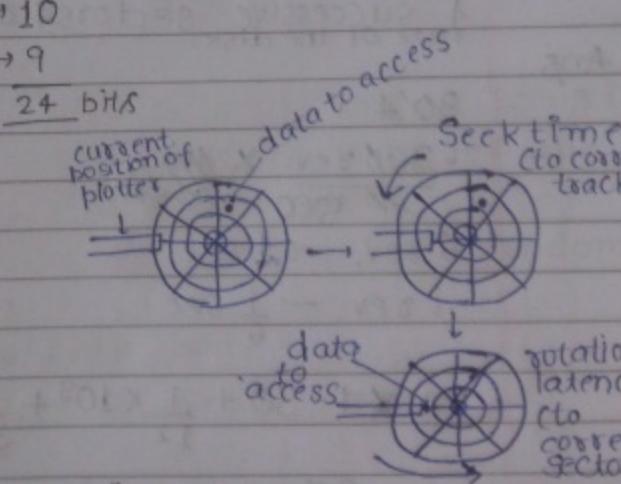
Q Consider a disk with 16 platters, each platter is divided into 2 surfaces, every surface has 1 K tracks & every track has 512 sectors & every sector can store 2 KB data, what is the capacity of disk?

$$\begin{aligned}
 \text{Ans. } & 16 \times 2 \times 2^{10} \times 2^9 \times 2 \times 2^{10} \text{ B} \\
 & = 2^5 \times 2^{10} \times 2^9 \times 2^1 \text{ B} \\
 & = 2^{35} \text{ B} \\
 & = 32 \text{ GB}
 \end{aligned}$$

Q. How many bits are req. to identify a sector in the above configuration of disk

Ans. 24 bits.

- to identify platter \rightarrow 4
- to identify surface \rightarrow 1
- to identify track \rightarrow 10
- to identify sector \rightarrow 9



Disk I/O Operation :-

- Seek Time
- Rotational Latency
- Transfer time
- 4 → Transfer rate

- **Seek Time :-** The amount of time taken to move the read/write head from its current position to desired track.
- The read/write heads ~~can only~~ can't be outside the platter.
- **Rotational Latency :-** The amount of time taken to rotate the track when the read/write head comes

Cosmos

to exact position (exact sector within a track). It is considered as $\frac{1}{2} \times$ rotation time.

Transfer Time :- The amount of time taken to transfer the required data. The transfer time depends on the total size of the track & rotational rate of the disk.

• Transfer Rate :- The no. of bytes transferred for unit time is called as transfer rate.

Q. If a disk system has an avg. seek time of 30 ns & rotational rate of 360 rpm. Each track of the disk has 512 sectors each of size 512 B. What is the time taken to read 4 successive sectors & also compute data transfer rate of the disk.

Ans.

30/4

$$6360 \text{ rev. } \times 60 \text{ min.} \\ = 280 \text{ sec. } .60 \text{ sec.}$$

$$6 \text{ rev. } \rightarrow 1 \text{ sec.}$$

$$1 \text{ rev. } \rightarrow \frac{1}{6} \text{ sec.}$$

$$512 \text{ sector } \rightarrow \frac{1}{6} \text{ sec.}$$

$$1 \text{ sector } \rightarrow \frac{1}{6 \times 2^9} \text{ sec}$$

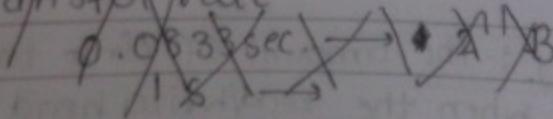
$$4 \text{ sector } \rightarrow \frac{1}{6 \times 2^9} \times 4 = \frac{1}{3 \times 2^8} \text{ sec}$$

$$30 \times 1 + 30 + \frac{1}{12} \times 10^9 + \frac{2}{3 \times 2^8}$$

$$= \text{negligible } 30 \times 10^{-9} + \frac{1}{12} + \frac{2}{3 \times 2^8}$$

$$\approx 0.0833 \text{ sec. } \xrightarrow{\text{negligible}}$$

Transfer rate



Cosmos

classmate

Date _____
Page _____

Transfer time:-

In one rotation time → can read entire track data
time required? → to read the req. data

1 rotation time:-

$$360 \rightarrow 60 \text{ sec} : \\ 1 \rightarrow \frac{60}{360} \text{ sec} \rightarrow \frac{1}{6} \text{ sec.}$$

Entire track size:-

$$512 \times 512 \text{ B} = 2^9 \times 2^9 \text{ B} = 2^{18} \text{ B} = 256 \text{ KB}$$

$$256 \text{ KB} \rightarrow \frac{1}{6} \text{ sec.}$$

$$1 \text{ B} \rightarrow \frac{1}{6 \times 256 \times 2^{10}}$$

$$2 \text{ KB} \rightarrow \frac{1}{6 \times 256 \times 2^{10}} \times 2 \times 2^{10} \rightarrow$$

$$1 \text{ sec.} \rightarrow 6 \text{ rev.}$$

$$1 \text{ sec.} \rightarrow 256 \text{ KB} \times 6$$

$$\boxed{1536 \text{ KBPS}}$$

Q. If a disk system has an avg. seek time of 60 ns & rotational rate of 3600 rpm. Each track of the disk has 256 sectors each of size 2KB.

(a) What is the approx. time req. to read 1200 random sectors?

(b) What is the data transfer rate?

Ans. 60 ns +

$$(60 \times 10^{-9} \text{ s} + \frac{1}{60} \text{ s} \times 1 + \frac{1}{60 \times 256}) \times 1200 \\ (0.00833) \times 1200$$

$$\approx 10 \text{ sec.}$$

$$3600 \text{ rev.} \rightarrow$$

$$60 \text{ rev.} \rightarrow 1 \text{ sec.}$$

$$1 \text{ rev.} \rightarrow \frac{1}{60} \text{ sec.}$$

$$256 \text{ sec.} \rightarrow \frac{1}{60} \text{ sec.}$$

$$1 \text{ sector} \rightarrow \frac{1}{60 \times 256} \text{ sec.}$$

$$1 \text{ rev.} \rightarrow \frac{1}{60} \text{ sec.}$$

$$256 \text{ sectors} \rightarrow \frac{1}{60} \text{ sec.}$$

$$30 \text{ MBPS} \leftarrow 30720 \text{ KBPS} \leftarrow 512 \times 2^{10} \times 60 \text{ B} = \frac{2^{19} \times 60 \text{ B}}{2^{19} \times 60 \text{ B}} = \frac{270 \text{ MB}}{6 \text{ MB}}$$

Cosmos

classmate

Date _____
Page _____

- Q. An application loads 100 libraries at startup. Load each library requires exactly 1 disk access. The seek time of the disk to a random location is 10 msec, the rotational speed of disk is 6000 rpm. If all 100 libraries are loaded from random location on disk, how long does it take to load all libraries (the time to transfer data from disk block once head has been positioned at the start of the block may be neglected)?

Ans.

$$100 \left(10 \times 10^{-3} + \frac{1}{2 \times 100} \right)$$

Transfer
time is
neglected.

$$\frac{6000 \text{ rev.}}{1 \text{ rev.}} \times \frac{1}{100}$$

$$100 \left(10^{-2} + \frac{1}{2 \times 100} \right)$$

$$1 + 0.5 = [1.5 \text{ sec.}]$$

- Q. Consider the Disk with below specifications:-

- (a) No. of surfaces = 8
 - (b) Inner diameter = 4 cm
 - (c) Outer diameter = 12 cm
 - (d) Inner track distance \rightarrow 0.2 mm [distance b/w any two consecutive concentric tracks]
 - (e) No. of sectors per track \rightarrow 20
 - (f) Sector size \rightarrow 4 KB
 - (g) Rotational Rate \rightarrow 3600 rpm
- (a) What is the capacity of disk?
- (i) 512 MB
 - (ii) 128 MB
 - (iii) 64 MB
 - (iv) 14 B
- (b) What is data transfer?

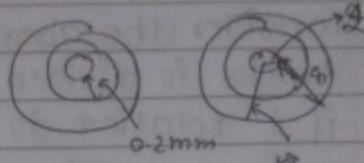
Cosmos

classmate

Date _____

Page _____

$$\begin{aligned}
 a) & 8 \times 900 \times 20 \times 4 \text{ KB} \\
 & 8 \times 2 \times 8 \times 10^3 \text{ KB} \\
 & 128 \times 10^3 \text{ KB} \\
 & 128 \text{ MB}
 \end{aligned}$$



$$\# \text{ of tracks} = \frac{4 \text{ cm}}{0.2 \text{ mm}} = \frac{10}{1 \text{ cm}}$$

360° rev. \rightarrow 60 sec.

$$1 \text{ rev.} \rightarrow \frac{1}{60} \text{ sec.}$$

1 sec. \rightarrow 60 rev.

1 sec. \rightarrow 60×20 sectors

$$\rightarrow 60 \times 20 \times 4 \text{ KB}$$

$$\rightarrow 1200 \times 4 \text{ KB}$$

$$\rightarrow 1.2 \times 4 \text{ MB}$$

$$\rightarrow 4.8 \text{ MB}$$

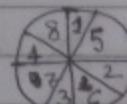
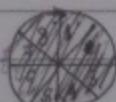
4.8 MBPS

Disk Interleaving -



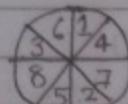
No interleaving disk

1 rotation



Single interleaving disk

2 rotations req. to read all 8 sectors of the disk



Double interleaving disk

3 rotations req. to read all 8 sectors of the disk.

But exactly 2.75 rotations

★ If there is slow

★ If it takes more time to read the data

Cosmos

- Q. Consider the above double interleaving disk who's tracks is divided into 8 sectors each of size 2 KB . If $\frac{1}{2}$ rotation is req. to place read/write head at start of the sector, seek time = 30 ms & rotational rate is 3600 rpm .
- How much time is req. to read all 8 sectors?
 - What is the data transfer rate.

Ans. $360 \text{ rev.} \rightarrow 60 \text{ sec.}$ $8 \text{ sectors} \rightarrow \frac{1}{60} \text{ sec}$
 $1 \text{ rev.} \rightarrow \frac{1}{60} \text{ sec.}$ 1 sec.

~~$\frac{1}{120} \text{ sec.} \times 3 \times 10^{-2} \text{ s} + 2.75 \times \frac{1}{60} \text{ sec.} + 8 \times \frac{1}{60} \text{ sec.}$~~

$$= 0.03 + 0.0229 + 0.0167$$

$$= 0.069 \text{ sec. } \times \rightarrow \text{Wrong.}$$

(b) 1 sector \rightarrow 1

$2.75 \text{ rev.} \rightarrow 8 \text{ sectors}$

$2.75 \times \frac{1}{60} \text{ sec.} \rightarrow 16 \text{ KB}$

$349.1 \text{ KBPS} \checkmark \rightarrow \text{correct.}$

Cosmos

Answer:- Seek time + Rotational Latency + Transfer time
(a) $30 \text{ ms} + 0.5 \times \text{Rotation time} + 2.75 \times \text{Transfer time}$

$$= 30 \text{ ms} + 0.5 \times \frac{1}{60} + \frac{2.75}{60}$$

$$= 30 \text{ ms} + 8.33 \text{ ms} + 45.83 \text{ ms} \\ \approx 84 \text{ ms}$$

(b) ~~A \times ex. A $\frac{Y}{60}$ sec.~~

3/18/09 HHR

