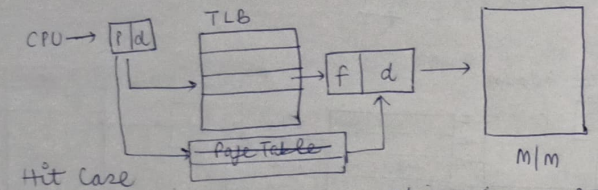
Translation Lookaside Buffer (TLB)

Numerical:-

A paging scheme using TLB. TLB access time 10 ns and main memory accen time takes 50 ns. What is effective memory accen time (in ns) if TLS hit ratio is 90% and there is no page fault



Hit Case

EMAT = Hit (TLB + M/m) + Miss (TLB + PT + M/M)
(M/M)

1-Hit
-10%

90%. (10+50) + 10%. (10+50+50)

.9 .(60) + .1 (110)

= 54+11 = 65 ns

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

- Page Replacement Algorithm

1) FIFO Page Replacement
2) Optimal Page Replacement
3) LRU Page Replacement

1) FIFO Page Replacement

- The simplest page replacement algo is a FIFO algorithm.
- A FIFO replacement algorithm associates with each page the time when page was brought into memory, When a page must be replaced the oldest page is chosen.

We can create FIFO queue to hold all pages in memory. We replace the page at head of queue.

→ FIFO page replacement algo is easy to understand & program However its performance is not always good. The page replaced may be initialization module that was used a long time ago & is no longer needed

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Page Fault :- Jab b CPU page ko main memory

f_3			1	1	1	1	0	0	0	3	3	3	3	2	2
f_2		0	0	0	0	3	3	3	2	2	2	2	1	1	1
f_1	7	7	7	2	2	2	2	4	4	4	0	0	0	0	0
	*	*	*	*	Page hit	*	*	*	*	*	*	Page hit	*	*	Page Hit

Reference string: ~~7~~, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 1, 2, 0

Hit = 3

Page Fault = 12

Page Miss

Hit ratio / ~~Miss ratio~~ = $\frac{\text{No of hits}}{\text{Total no of reference}} = \frac{3}{15} \times 100 = 20\%$

Page Miss = $\frac{12}{15} \times 100 = 80\%$

→ Belady's Anomaly

- FIFO is suffering from Belady's Anomaly
- It states that if we increase the frames the no of page faults should decrease but this is not happening page faults are increasing.

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

f_3			3	3	3	2	2	2	2	2	4	4				
f_2		2	2	2	1	1	1	1	1	3	3	3				
f_1	1	1	1	4	4	4	5	5	5	5	5	5				
	*	*	*	*	*	*	*	Hit	Hit	*	*	Hit				

Reference string :- 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

Hit = 3

Page Fault = 9

f_4				4	4	4	4	4	4	3	3	3			
f_3			3	3	3	3	3	3	2	2	2	2			
f_2		2	2	2	2	2	2	1	1	1	1	5			
f_1	1	1	1	1	1	1	5	5	5	5	4	4			
	*	*	*	*	Hit	Hit	*	*	*	*	*	*			

Hit = 2

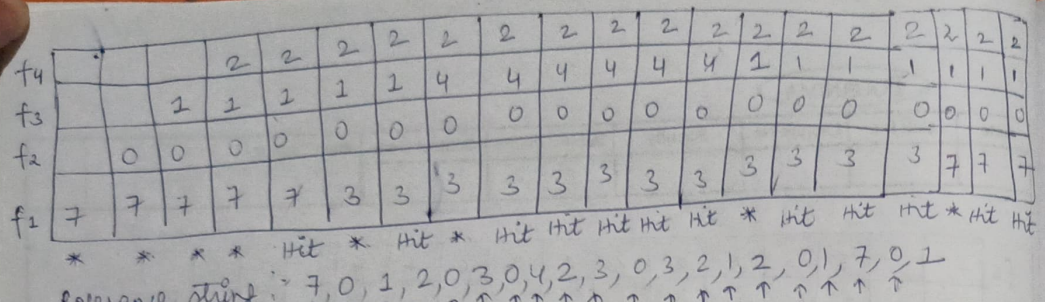
Page Fault = 10

Optimal Page Replacement

- Replace the page that will not be used for the longest period of time.
- An optimal page replacement algorithm has the lowest page fault rate of all algorithms will never suffer from Belady's anomaly.

Main Ideas, Questions & Summary:

Library / Website Ref.:-



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

Hit :- 12

Page Fault :- 8

Hit Ratio : $\frac{12}{20} \times 100$

Page Miss/ Page Fault : $\frac{8}{20} \times 100$

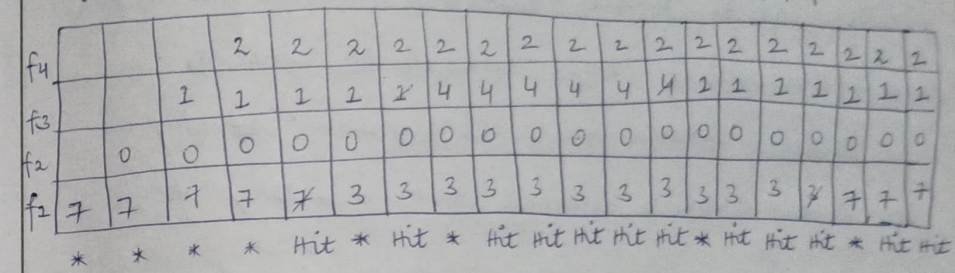
LRU Least Recently Page Replacement.

- Replace the least recently used page in past
- LRU replacement associates with each page the time of that page's last use. When a page must be replaced LRU chooses the page that has not been used for longest period of time. This strategy is optimal page replacement algorithm looking backward in time than forward.
- LRU policy is often used as page replacement algo is considered to be good.

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

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



Page Fault = 8 Hit = 12

MRU Most Recently Used

(Replace the most recently used page in past)

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Demand Paging

1) Only loads pages are demanded by executing process.
2) As there is more space in main memory, more processes can be loaded reducing context switching time which utilizes large amounts of resources.
3) Less loading latency occurs at program startup as less information is accessed from secondary storage
4) Memory management with page replacement algo becomes slightly more complex

Pure Paging

1) All pages are pre-loaded
2) Less processes can be loaded in memory context switching time.
3) More loading latency at program startup.
4) No need to memory management with page replacement algo.

POORNIMA

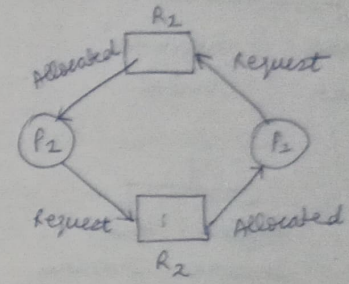
Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

DEADLOCK

If two or more process are waiting on happening of that event which never happens then we say these processes are involved in deadlock then that state is called deadlock

Necessary condition for Deadlock

1) Mutual Exclusion
2) No Premption
3) Hold & Wait
4) Circular Wait



1) Mutual Exclusion :- Each resource is either currently assigned to exactly one process or is available.
2) No Premption :- Resource previously granted cannot be forcibly taken away from a process. They must be explicitly released by process holding.
3) Hold & Wait :- Processes currently holding resource granted earlier can request new resources

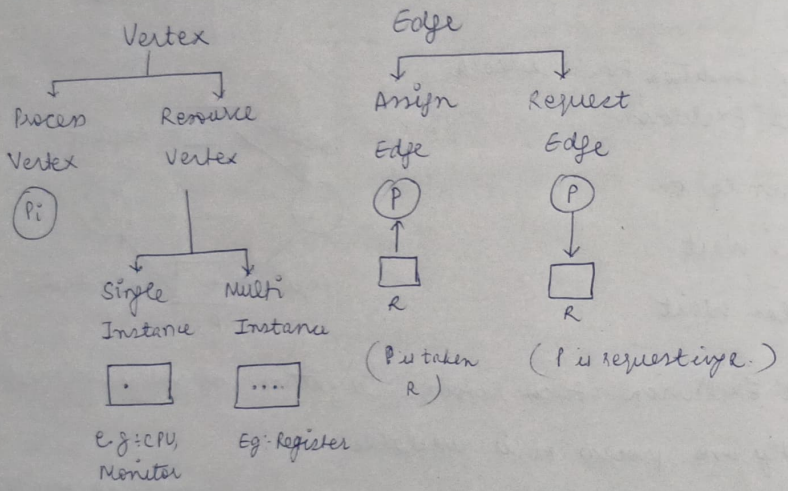
Main Ideas, Questions & Summary:

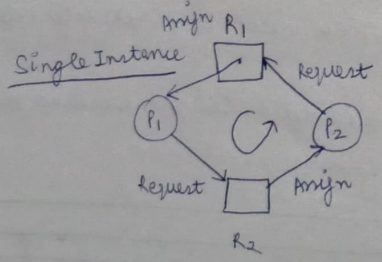
Library / Website Ref.:-

4) Circular wait condition

There must be circular chain of two or more process each of which is waiting for resource held by next number of chain

Resource Allocation Graph (RAG)





1) Circular wait

- P_1 is holding R_1 and waiting for R_2 similarly. P_2 is holding R_2 and waiting for P_1
- To execute P_1 want R_2 similarly P_2 want R_1 Both are waiting for one - one source no one will fulfill their resources That's why there is deadlock situation.

	Allocate		Request	
	R_1	R_2	R_1	R_2
~~P_1~~	1	0	0	1
P_2	0	1	1	0

R_1 R_2

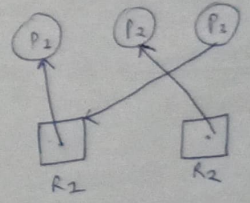
Availability (0, 0)

Terminated

	Allocate		Request	
	R_1	R_2	R_1	R_2
× P_2	1	0	0	0
× P_2	0	1	0	0
✓ P_3	0	0	1	1

Availability (0, 0)

1 0
0 1
1 1



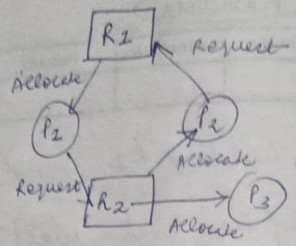
1) No Deadlock
2) Acyclic

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Multi Instance RAG

	Allocate		Request	
	R_1	R_2	R_1	R_2
× P_1	1	0	0	1
P_2	0	1	1	0
× P_3	0	1	0	0



Availability (0,0)

R_1	R_2	
0	1	
0	1	Current Availability
1	0	
1	1	Current Availability

(P_3, P_1, P_2)

1) There is no deadlock
2) If there is cycle in multi instance there is no deadlock

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Various Methods to handle deadlocks

1) Deadlock ~~Avoidance~~ Ignorance (Ostrich Method)
   - Just ~~avoid~~ ignore the deadlock.
   - Our majority of operating system ~~use~~ like linux, windows etc both use strategy to ignore deadlock it is used widely.
   - ~~Dea~~ Programmer think deadlock occur very rare why we write full code to remove deadlock which will affect the performance or speed will degrade.

2) Deadlock Prevention
   The normal phrase is prevention is better than cure.
   The time when the deadlock occurs we're trying to find solution the better way is before the deadlock occur we try to find solution
   Four condition to ~~prevent~~ have deadlock :
   1) Mutual Exclusion
   2) No Prempion
   3) Hold & Wait
   4) Circular wait
   → If one ~~of~~ the condition is false then deadlock will be prevented

Main Ideas, Questions & Summary:

Library / Website Ref.:-

3) Deadlock Avoidence

(Current Availability I Remaining Need

Banker's Algorithm

Total A=10, B=5, C=7

Process	Allocation (CPU) A	Allocation (Memory) B	Allocation (Printer) C	Max Need A	Max Need B	Max Need C	Available A	Available B	Available C	Remaining Need (Max-Allocation) A	Remaining Need (Max-Allocation) B	Remaining Need (Max-Allocation) C
							3	3	2			
P_1	0	1	0	7	5	3				7	4	3
P_2	2	0	0	3	2	2	5	3	2	1	2	2
P_3	3	0	2	9	0	2				6	0	0
P_4	2	1	1	4	2	2	7	4	3	2	1	1
P_5	0	0	2	5	3	3	7	4	5	5	3	1
	7	2	5				7	5	5			
							10	5	7			

Safe Sequence

P_2 → P_4 → P_5 → P_1 → P_3

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

4) Deadlock Detection

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Scheduling Algorithms

Pre Emptive

- SRTF (Shortest Remaining Time First)
- LRTF (Longest Remaining Time first)
- Round Robin
- Priority Based

Non Pre Emptive

- FCFS (first Come First Serve)
- SJF (Shortest Job first)
- LJF (Longest Job First)
- HRRN (Highest Response Ratio Next)
- Multi level Queue

CPU Scheduling

- Arrival Time :- The time at which process enter the ready queue
- Burst Time :- Time required by a process to get execute on CPU (Duration)
- Completion Time :- The time at which process complete its execution.
- Turn Around Time :- {Complete Time - Arrival Time}
- Waiting Time :- {Turn Around Time - Burst Time}
- Response Time :- {(Time at which process get CPU first time) - (Arrival Time)}

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

FCFS First Come First Serve

Process No	Arrival Time	Burst Time	Completion Time	TAT (CT-AT)	WT (TAT-BT)	RT
P_1	0	2	2	2-0=2	2-2=0	0-0=0
P_2	1	2	4	4-1=3	3-2=1	2-1=1
P_3	5	3	8	8-5=3	3-3=0	5-5=0
P_4	6	4	12	12-6=6	6-4=2	8-6=2

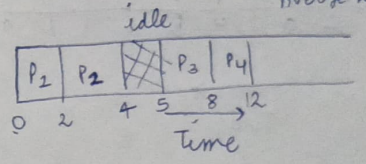
Criteria : "Arrival Time"

Mode : "Non Preemptive"

Average TAT = $\frac{14}{\text{Total Process}} = \frac{14}{4} = 3.5$

Average WT = $\frac{3}{4} = .75$

Gantt Chart



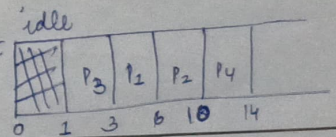
Shortest Job First (SJF)

Process No	Arrival Time	Burst Time	Completion Time	TAT (CT-AT)	WT (TAT-BT)	RT
~~P_1~~	1	3	6	6-1=5	5-3=2	3-1=2
P_2	2	4	10	10-2=8	8-4=4	6-2=4
~~P_3~~	1,	2	3	3-1=2	2-2=0	1-1=0
P_4	4	4	14	14-4=10	10-4=6	10-4=6

Criteria :- "~~Burst Time~~"

Mode : "Non Preemptive"

Gantt Chart



Average TAT = $\frac{25}{4} = 6.25$ Average WT = $\frac{12}{4} = 3$

Main Ideas, Questions & Summary:

Library / Website Ref.:-

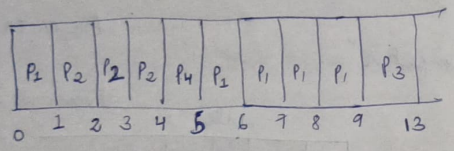
Shortest Remaining Time First (SRTF)

Criteria : Burst Time
Mode : Preemptive

TAT = CT - AT
WT = TAT - BT

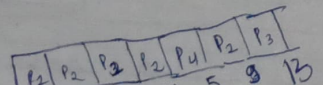
Process	Arrival	Burst	Completion	TAT	WT	RT
~~P_1~~	0	~~5 4 3 2 1 0~~	9	9-0=0	~~0~~ 4	~~0~~ 0
~~P_2~~	1	~~3 2 1 0~~	4	4-1=3	0	0
P_3	2	~~4 3~~	13	13-2=11	7	7
~~P_4~~	4	1 0	5	5-4=1	0	0

Gantt Chart



Average TAT = $\frac{24}{4} = 6$ Average WT = $\frac{11}{4} = 2.75$

Average RT = $\frac{7}{4} = 1.75$



POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Round Robin Algo

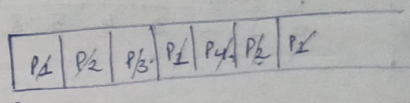
Criteria : "Time Quantum"
Mode "Inemptive"

TAT = CT - AT
WT = TAT - BT

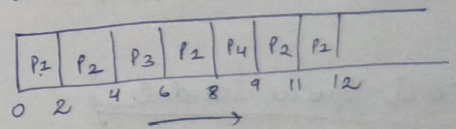
Process	Arrival	Burst	Completion	TAT	WT	RT
P_1	0	~~5 3 1~~	12	12	7	0
~~P_2~~	1	~~4 2 0~~	11	10	6	1
~~P_3~~	2	~~2 0~~	6	4	2	2
~~P_4~~	4	1 0	9	5	4	4

Gantt Chart

Ready Queue



Running Queue



Ready

Context switching.

Context switch = 6

Main Ideas, Questions & Summary:

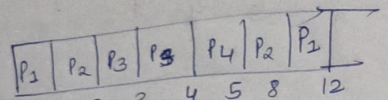
Library / Website Ref.:-

Priority Scheduling

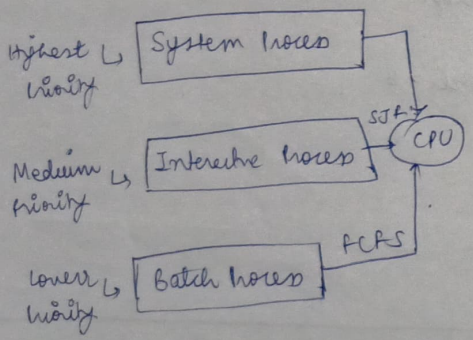
Criteria :- "Priority"
Mode = "Preemptive"

Proco Priority	Process No	Arrival	Burst	Completion	TAT	WT
10	P_1	0	54	12	12	7
20	P_2	1	43	8	7	3
30	P_3	2	Zero	4	2	0
40	P_4	4	10	5	1	0

Higher the no
Higher the priority



Multilevel Queue Scheduling

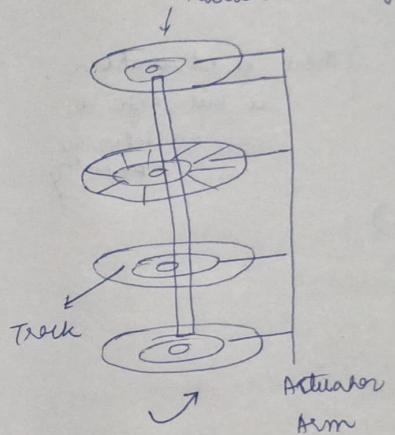


POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Disk Architecture

Platters → Surface → Track → Sector → Data



Disk Size : Platters × Surface × Track × Sector × Data

1K = 2^{10}
1M = 2^{20}
1G = 2^{30}
1T = 2^{40}

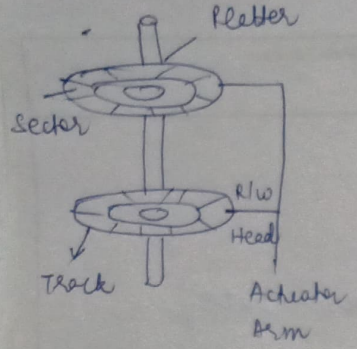
**

Disk Access Time

1) Seek Time :- Time taken by read write head to reach desired track.
2) Rotation Time :- Time taken for one full rotation (360°)
3) Rotational Latency :- Time taken to reach to desired sector (half of rotation time)
4) Transfer Time :- $\frac{\text{Data to be transfer}}{\text{Transfer Rate}}$, Transfer Rate OR Disk Rate { No of heads × Capacity × No of tracks of one rotation in one zone

Main Ideas, Questions & Summary:

Library / Website Ref.:-



- Hard disk is collection of multiple platters

Platter
↓
Surface
↓
Track
↓
Sector

DISK SCHEDULING ALGORITHMS

{There is starvation in this algo so we relve by FCFS}

Goal:- To minimize the seek time

→ FCFS (First come First Serve)

→ SSTF (Shortest Seek Time First

→ SCAN

→ Look

→ CSCAN (Circular SCAN)

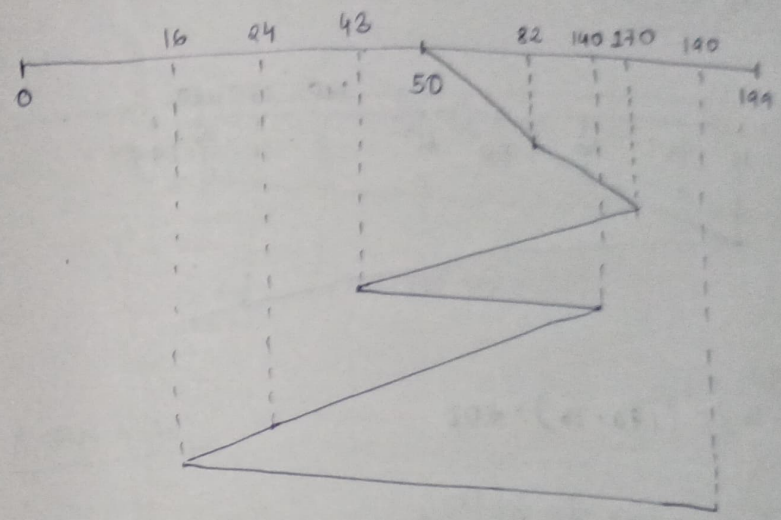
→ CLOOK (Circular look)

1) FCFS First Come First Serve)

Ques:- A disk contains 200 track (0-199) request queue contains track no 82, 170, 43, 140, 24, 16, 190 respectively current position of r/w head : 50. Calculate total no of tracks movement by R/w head.

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-



(170-50)+(170-43)+(140-43)+(140-16)+(190-16) = 642

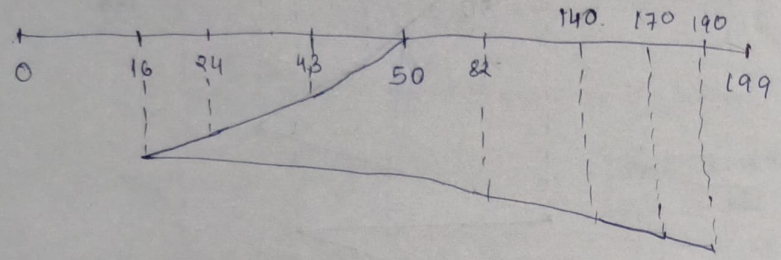
- There is no starvation in FCFS.
- It is simplest method of scheduling operation perform in order requested.
- Every process is serviced.

Main Ideas, Questions & Summary:

Library / Website Ref.:-

SSTF (Shortest Seek Time First)

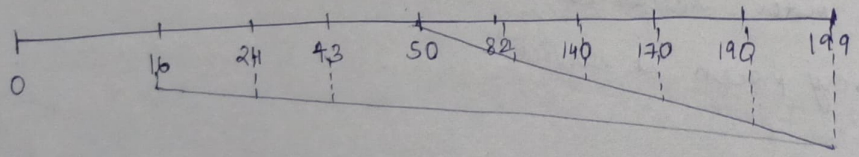
82, 170, 43, 140, 24, 16, 190



(50-16) + (190-16) = 208

- Starvation occurs in SSTF
- Complexity

SCAN Algorithm



(199-50) + (199-16) = 332

Disadvantage
→

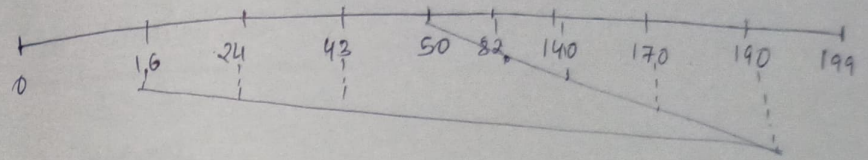
332+1 = 333 ms

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

82, 170, 43, 140, 24, 16, 190

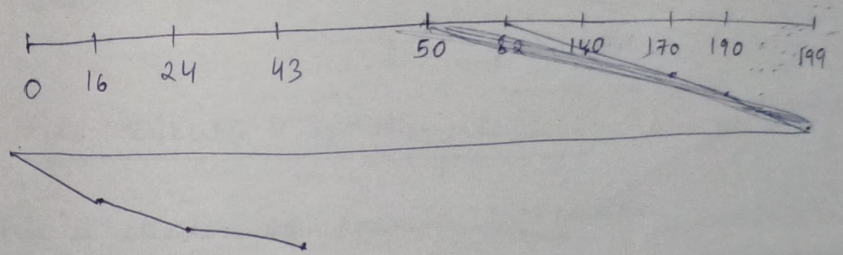
LOOK ALGO



(190-50) + (190-16) = 314

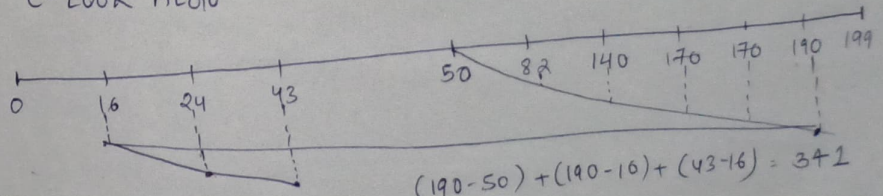
~~314+1 = 315 ms~~

CSCAN ALGO



(199-50) + (199-0) + (43-0) = 391

C-LOOK ALGO



(190-50) + (190-16) + (43-16) = 341

Main Ideas, Questions & Summary:

Library / Website Ref.:-

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

CHAPTER 5

~~FILE ATTRIBUTES & OPERATIONS IN Os~~

FILE SYSTEM IN Os

↓

Software

How file system stored data

How file system fetched data

User → file → folder / → file system
Directory

File Attributes & Operations In Os

→ File is collection of data or information

Main Ideas, Questions & Summary:

Library / Website Ref.:-

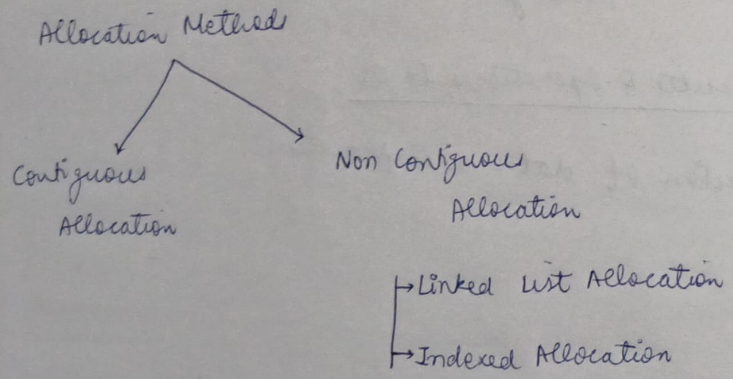
Operation

1) Creating
2) Reading
3) Writing
4) Deleting
5) Truncating
6) Repositioning

File Attributes

1) Name
2) Extension (Type)
3) Identifier
4) Location
5) Size
6) Modified date, Created date
7) Protection / Permision
8) Encryption, compremon

Allocation Methods In Operating System



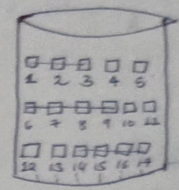
Advantages

1) Efficient Disk Utilization
2) Access faster

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

1) Continuous Allocation



Directory

File	start	length
A	0	3
B	6	5
C	14	4

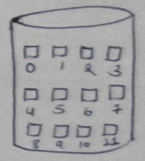
Advantages

1) Easy to implement
2) Excellent Read Performance

Disadvantages

1) Disk will become fragmented
2) Difficult to grow file

1) Linked List Allocation (Non Contiguous)



File	Start
A	2

Data / Pointer

Advantages

1) No external fragmentation
2) File size can increase

Disadvantages

1) Large seek time
2) Random access / Direct access Difficult
3) Overhead of pointer

Main Ideas, Questions & Summary:

Library / Website Ref.:-

2) Indexed File Allocation

Advantages

1) Support direct access
2) No External fragmentation

Disadvantages

1) Pointer overhead
2) Multilevel Index