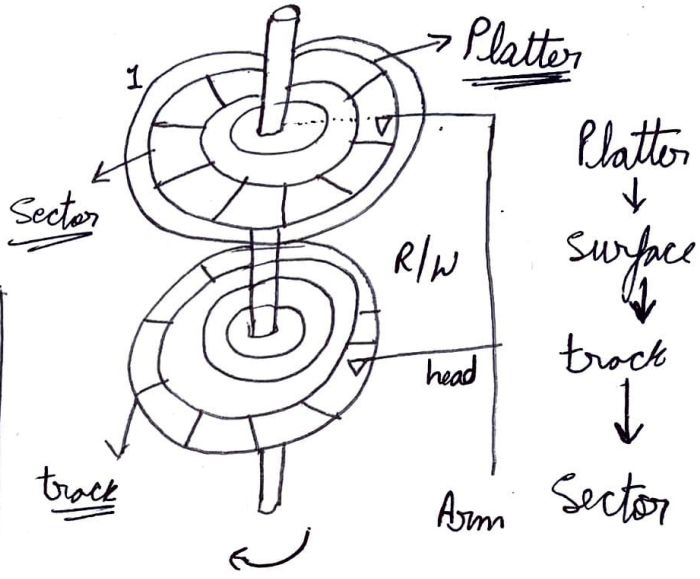
Disk Scheduling :-

- FCFS
- SSTF
- SCAN
- LOOK
- CSCAN (Circular SCAN)
- CLOOK (Circular LOOK)

• Rotational Latency - is the additional waiting time for the disk, to rotate the desired sector to the disk head.

Goal = To minimize the seek time.

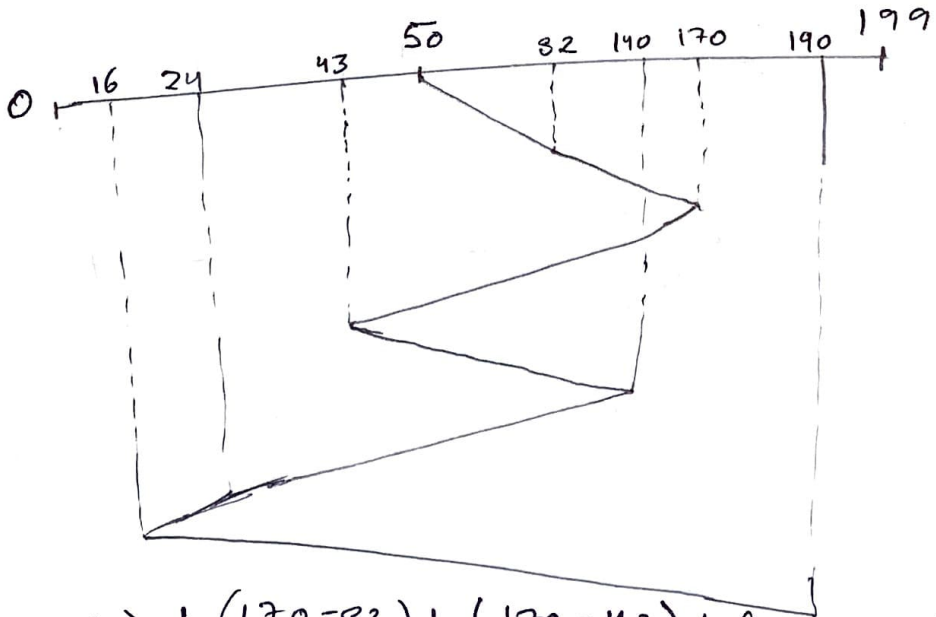
• Seek time → time taken to reach up to desired track



Drive rotates = 60 to 200 time/sec.

Q-1 A disk contains 200 tracks (0-199)
Request queue contains track no 82, 170, 43, 140, 24, 16, 190 respectively
Current Posⁿ of R/W head = 50. Calc. total no. of tracks movement by R/W head.

FCFS

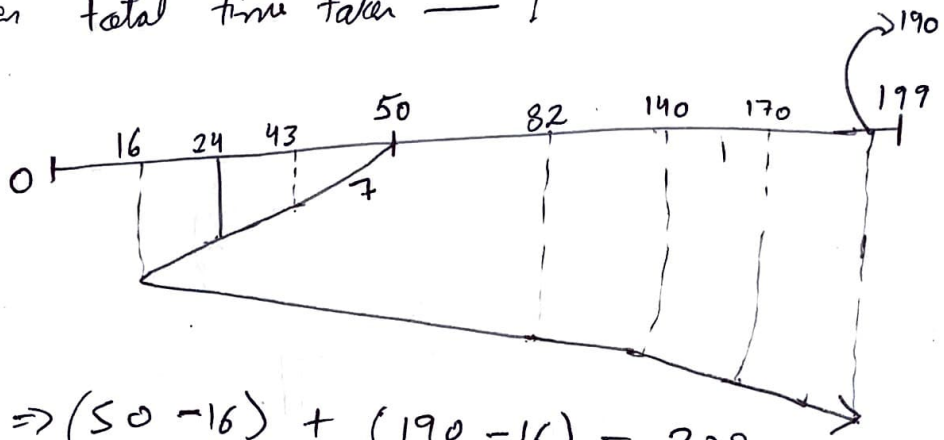


⇒ (82-50) + (170-82) + (170-43) + (140-43) + (140-24) + (24-16) + (190-16) = 642

Q- A disk contain 200 tracks (0 - 199) Request queue contains track no. 82, 170, 43, 140, 24, 16, 190 respectively. curr. Position of R/W Head = 50.

→ calculate total no. of tracks movemat by R/W head using Shortest Seek time first?

→ If R/W head takes 1ms to move from on track to another. then total time taken — ?



⇒ (50 - 16) + (190 - 16) = 208

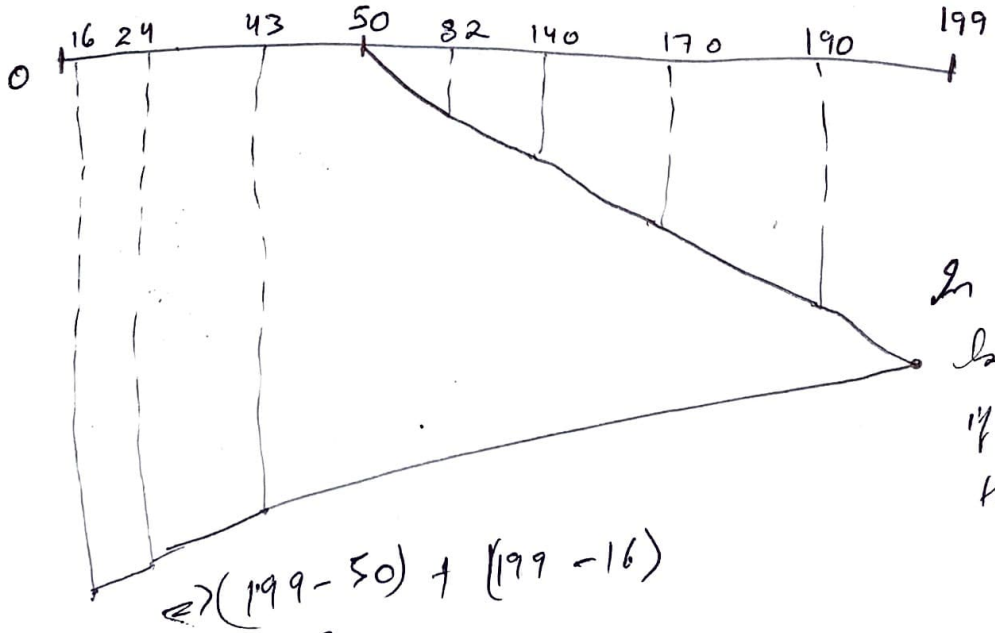
84

Mew. Prblm of Starvation occurs

Also, overhead generation means complexity ↑

Q- Above Same ! (→ calc total no of tracks movemat by R/W head using SCAN ? Also called elevator algo.

→ If R/W Head takes 1ms to move from on track to another then total time taken — ? ——→ A ⇒ 332 x 1

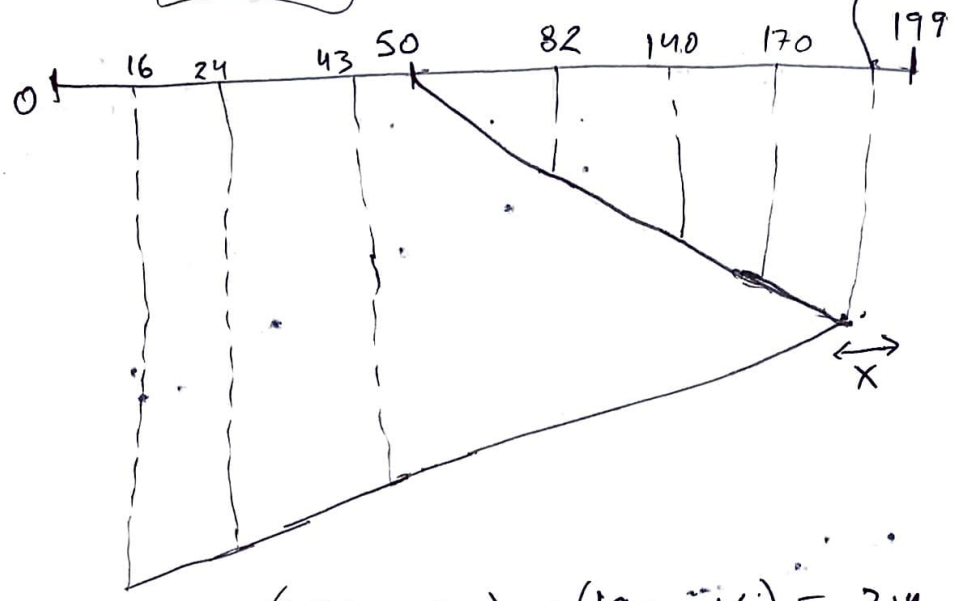


In this first move to last either 0 or 199 if move towards the Beg.

⇒ (199 - 50) + (199 - 16)

= 332.

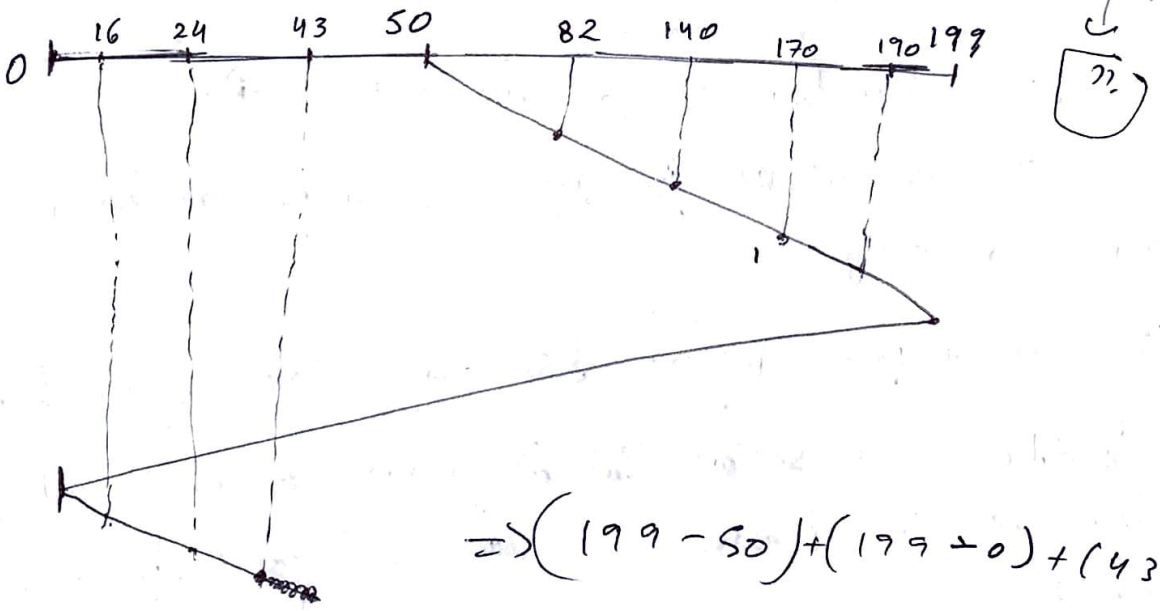
Q:- Above Same:- → Calc. total no of tracks movement by R/W head using **Look?**



If Given → Direction is towards large value..

→ (190 − 50) + (190 − 16) = 314

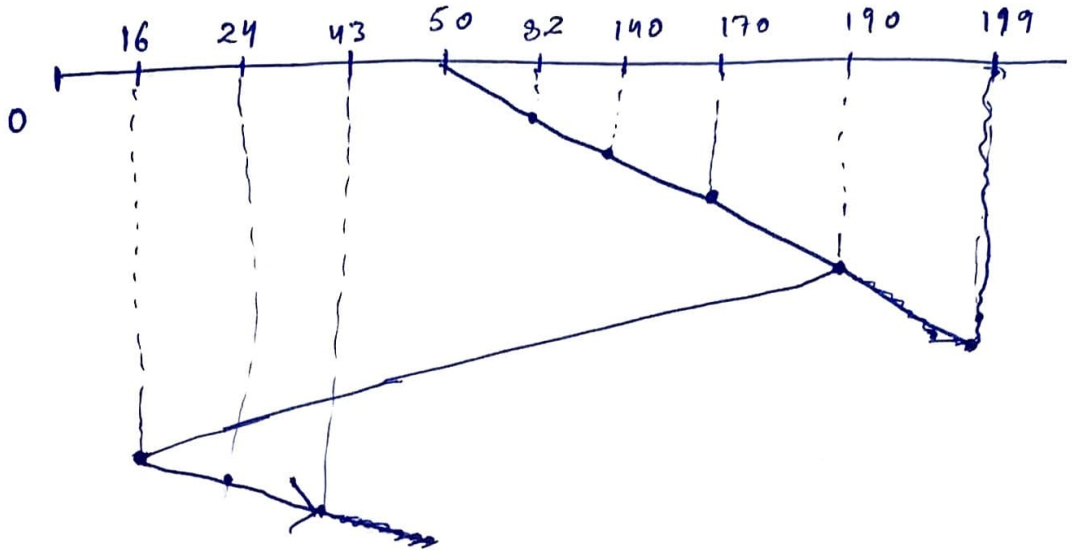
Q:- Above Same:- **C-SCAN**



⇒ (199 − 50) + (199 − 0) + (43 − 0)

= 391

Q- C-Look ; 82, 170, 43, 140, 24, 16, 190
C.P = 50



- ~~either~~ ~~SSTF~~ SCAN & C-SCAN performs better for System that place a heavy load on the disk..

Disk Management:-

- Its all about how we format our H.D.
- OS is responsible for disk formatting, booting & bad block recovery.

1) Disk formatting :- Before disk can store data, it must be divided into sectors that disk controller can read & write. This is known as low-level formatting or Physical formatting.

- A special data structure is added to each sector. It contains headers & data area, trailer Headers & trailer contains information like sector no: & CRC codes. (check weather sector is corrupted or not corrupted)

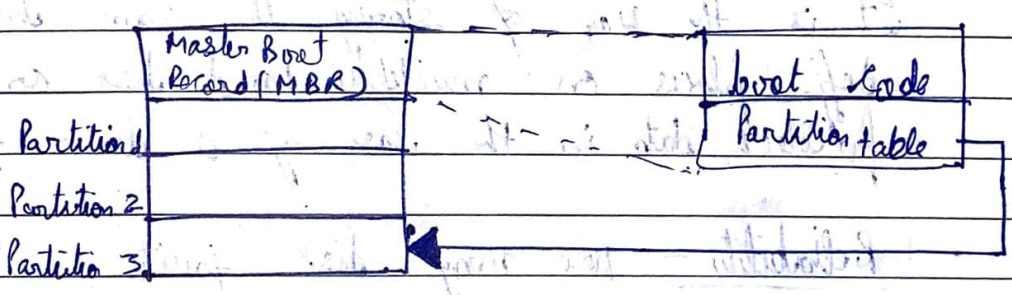
Page No.
Date :

O.S (Continue).

- O.S records its own data structure on the disk.
- To ↑ efficiency, file systems group blocks together into large clusters.

(2) Boot block :-

- When system is powered-on, the initial prgm called boot strap prgm is gets loaded to m/m which loads operating system.
- Bootstrap prgm is stored in ROM. It is a convenient location, as it needs no initialisation.



(3) Bad blocks :-

Sometimes due to movement of arm, controller & read-write head, the surface of disk may be corrupted, creating bad blocks on the platter of disk.

⇒ In MS DOS, "format" command performs logical formatting and scan the disk to find bad block.

• MS-DOS "chkdsk" ; Search & lock bad lock.

Page No.
Date :

If bad blocks is found, then in file Allocation table entry a special value is written in that block to tell OS to not to use this block.

* Controller replaces each bad sector logically with one of the spare sectors (sector sparing or forwarding).
• Each cylinder has a few spare sectors.

RAID :-

• Redundant array of inexpensive Disks.
(copy of disk).

It is the way of storing the same data in diff. places on multiple hard disks or SSD to protect data in the case of a drive failure.

• Reliability - How many disk faults can the system tolerate?
• Availability - how available is the system for actual use?
• Performance - How good is the response time? How high throughput!

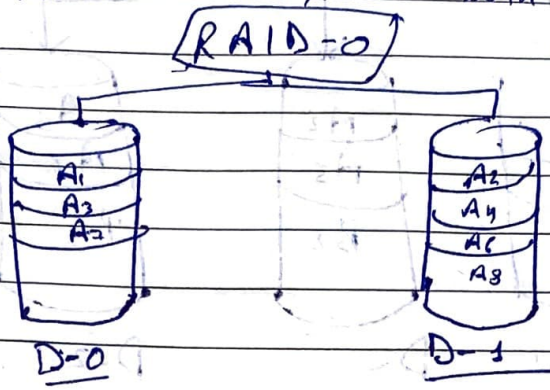
→ RAID-0 :- (Striping)

• Array of disks with block-level striping.
↑ performance

* So, adding redundancy almost always ↑ the reliability of the disk system.
* The most common way to add redundancy is (RAID).

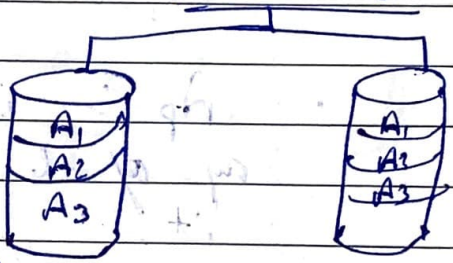
Page No.
Date :

- More throughput in this level as we can fly read or write data, As both disks are independent.

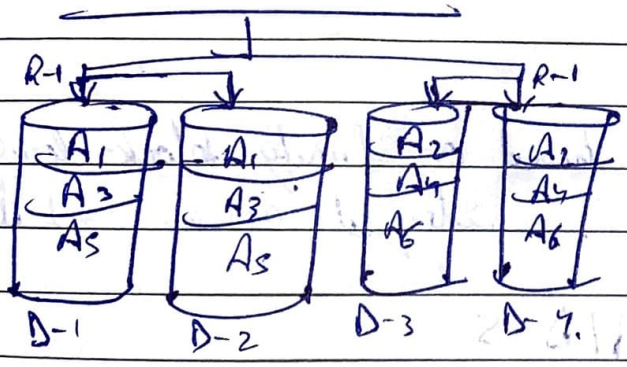


RAID-1 (Mirroring)

- It's main purpose is to secure data not performance. (as it keeps exact copy of the data).



RAID 1+0 :- (combination of 0).



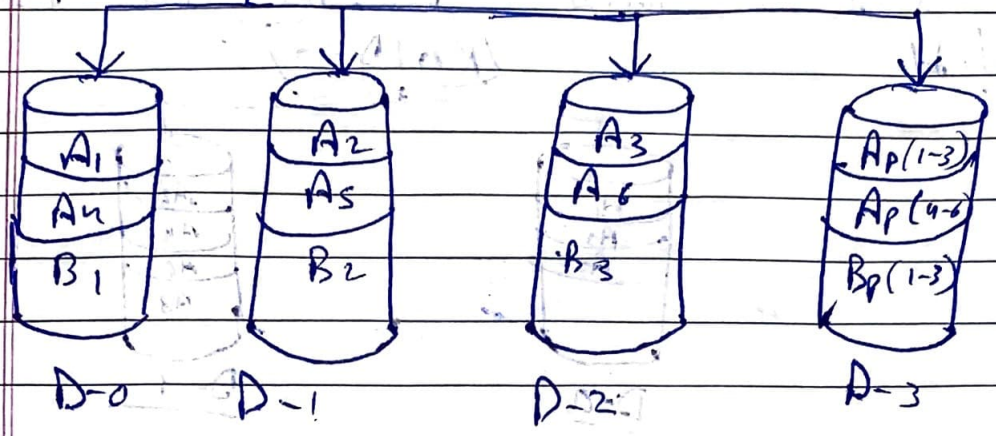
{ Mostly advantageous & used in web servers but it performs hdgs(a) as well as keeps data secure (1) }

RAID-2 :- (Bit-level Striping).

called a memory-style error correcting code organization

Page No.
Date :

→ RAID-3 :-

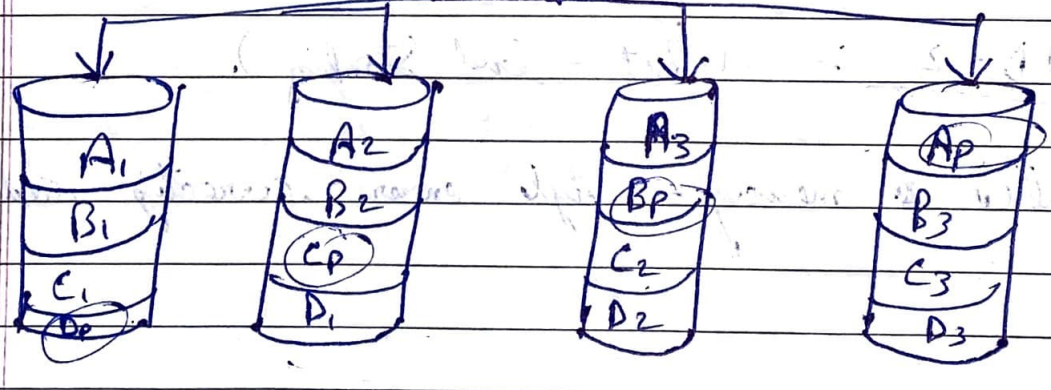


- In this data divided into blocks & capture in diff. disks
- Ap :- is parity disk, its factor is if any of the disk fail then still i can restore it.

→ RAI D-5 :-

- This level is based on parity block-level striping
- The parity info is striped across each drive.

RAID-5

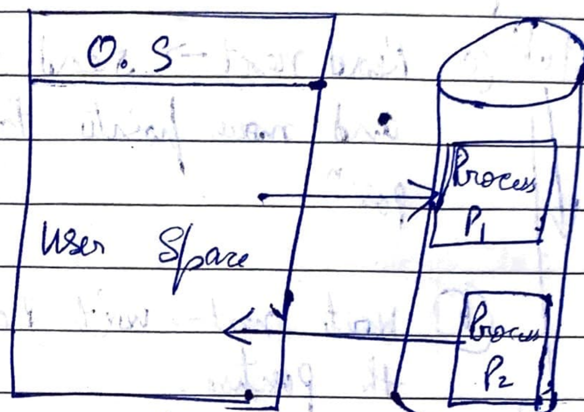


- Data availability ✓

Swap - Space Management :-

- It is a technique of removing a process from main memory and storing it into S.M and then bringing it back into M/M for continued execution.

- SS goal is to provide the best throughput.



Swapping of process

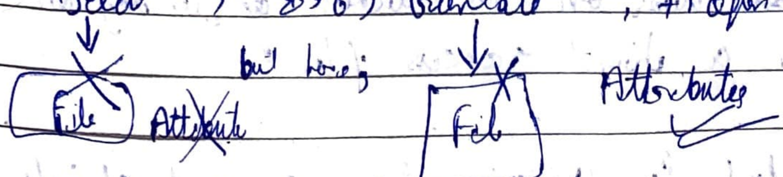
File System (in OS) :-
↓
Software

- Files Attributes :-

1) Name, 2) extension (type), 3) Identifier
4) Location, 5) Size, 6) Modified date ; Created dat
7) Protection / Permission, 8) Encryption, Compression

- File operations :

1) File is an abstract data type.
2) Create, 3) Write, 4) Read, 5) Reposition within file
5) Delete, 6) Truncate, 7) open (Fi), 8) close (Fr)



File access Methods :-

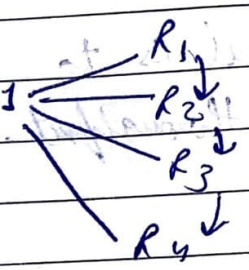
(1) Sequential Access :- (S.A)

- Info. in the file is proceeded in order.
- one record after the other
- editor & compilers usually access files in this fashion

operations

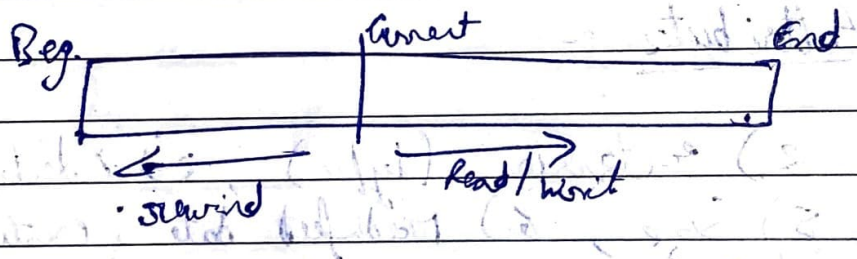
@ Read next → read record and move pointer to next posⁿ.

(b) Write next → write & advances the position.



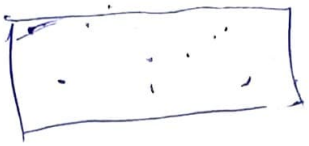
© Rewind – Moving back to earlier location.

(d) Skip n records — skip (2)



(2) Direct Access :- (D.A) / Random / Relative Access

- It is based on disk model. Since disks allow random access to any file block.
- No restrictions on the order of reading or writing for a direct access file.
- very useful for accessing large amount of info.



CP = Count pointer

Page No.

Date :

- file operations must be modified to include the block no. as a parameter

[S.A or D.A]

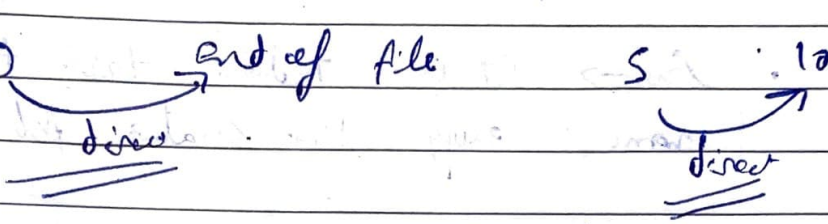
→ CP = 0

@ Read n → n is the block no.
(b) Write n → " " " "
© Jump to rewrite n → " " " "

read CP
CP = CP + 1
read next

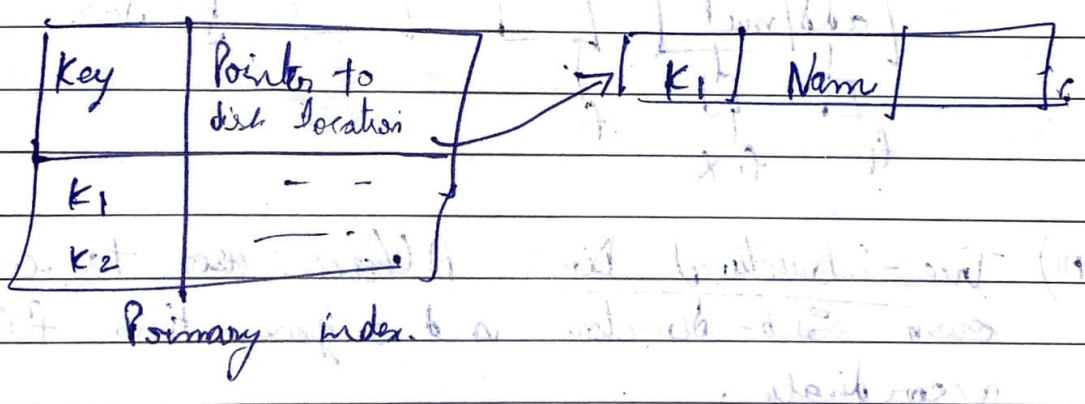
Write CP
CP = CP + 1
write next

used in db.



(d) query current record → used to return back to this record later.

(3) Indexed Access :- Indexed is created which contains a key field and pointers to the various blocks



File Directories :-

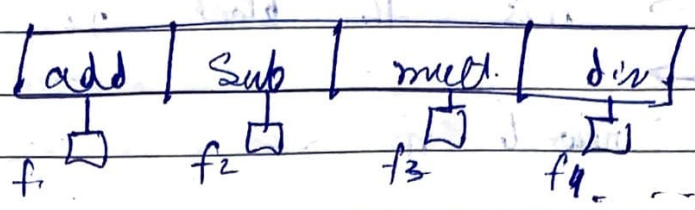
A physical disk can be broken up into multiple partitions, or mini-disks.

→ op.n on directories :

1) Search for a file, 2) Create new file, 3) Delet a file
4) list a directory, 5) Rename a file, 6) traverse file system.

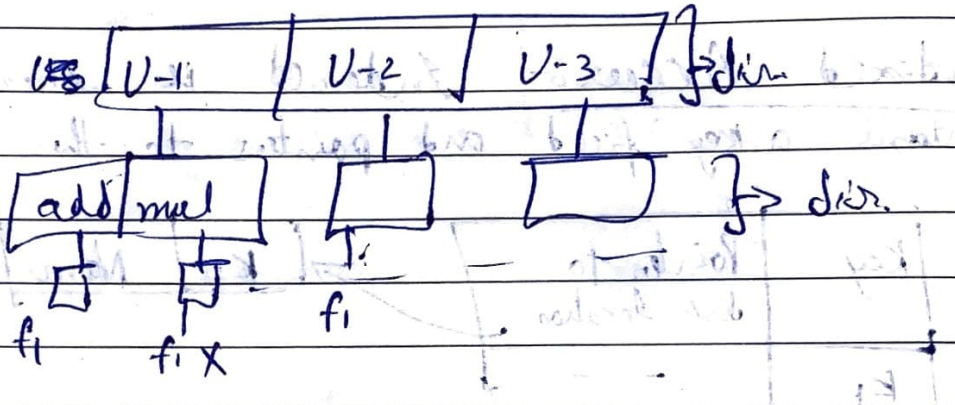
i) Single - level directory :- All files contained in same directory.

- each file must have unique name.



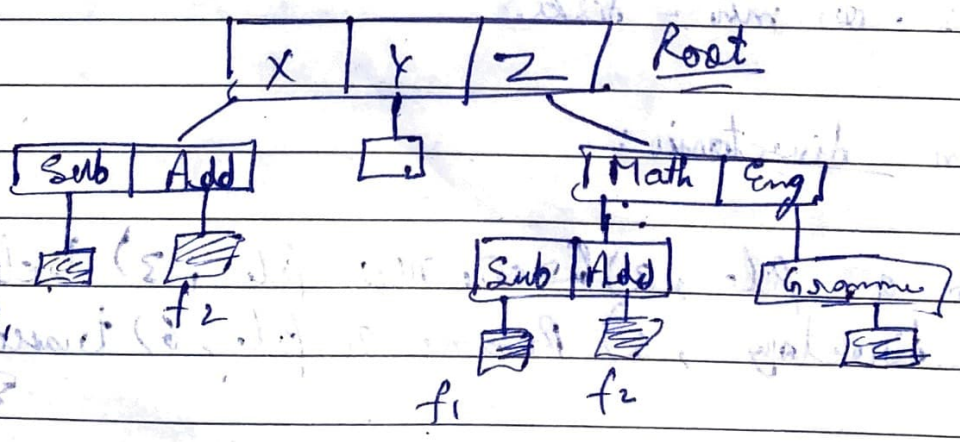
- Loss → it is tedious task to look for a unique name every time creating file.

ii) Two - level directory :- Creating a directory for each user.



iii) Tree - Structured Dir :- Allows user to create their own sub - directories and organize their files accordingly.

Path : It is the path (root) from the root through all the sub directories to specified file.

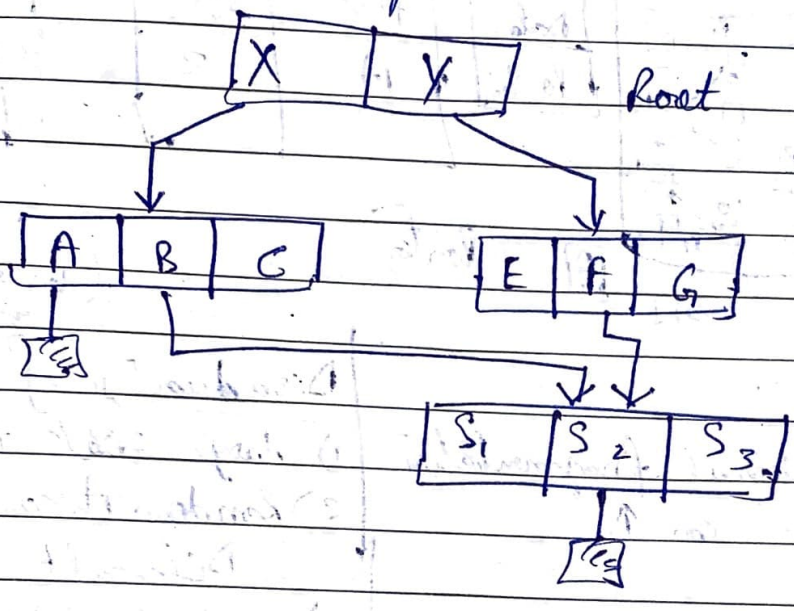


File allocation table (FAT)

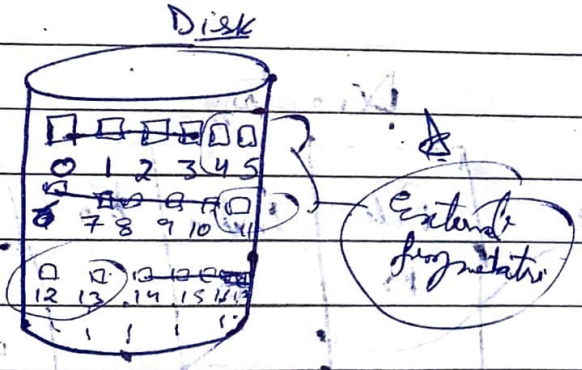
Page No.
Date :

iv) Acyclic Graph Directories : Useful when the same files need to be accessed in more than one place in the directory structure.

[Files are shared by more than one user / Process]



Contiguous Allocation :

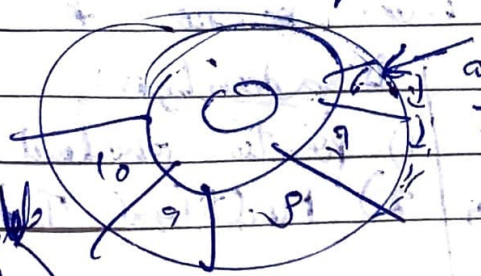


Directory

File	Start	Length
A	0	3
B	6	5
C	14	4

- Advantages :-
1) Easy to implement
2) excellent read performance

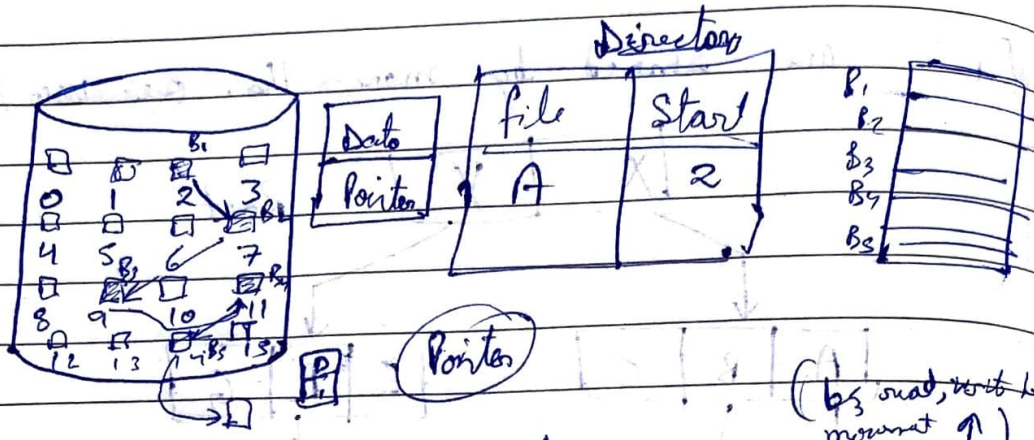
Disadvantages :
1) Disk will become fragmented
2) Difficult to grow file.
- Problem of external fragmentation



Page No.
Date :

Linked list Allocation :-

. It comes under the non-contiguous Allocation



(b3 read, write head movement ↑)

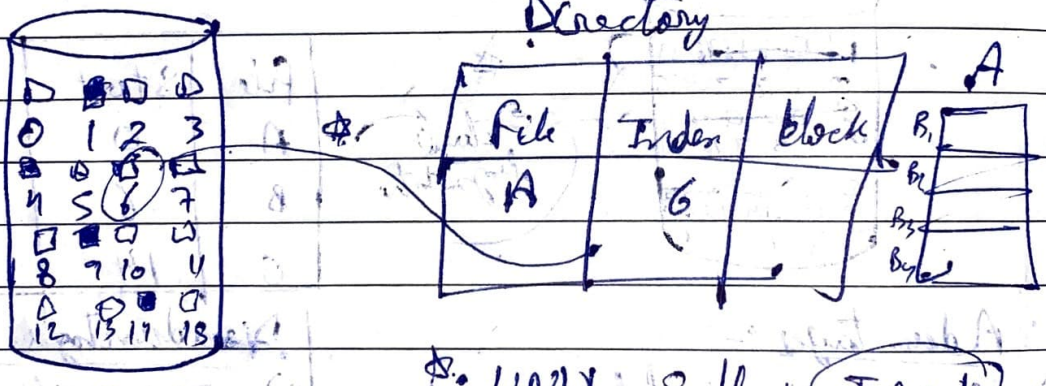
. Advantage :-

1) No external fragmentation
2) File size can ↑.

Disadvantage :-

1) Long seek time
2) Random Access / Direct Access Difficult
3) overhead of pointers

Indexed Allocation :-



H.D

* UNIX Support (I Node)

. Each file having its index no. ↑

. Advantage :-

1) Support direct Access
2) No external fragmentation

Disadvantages :

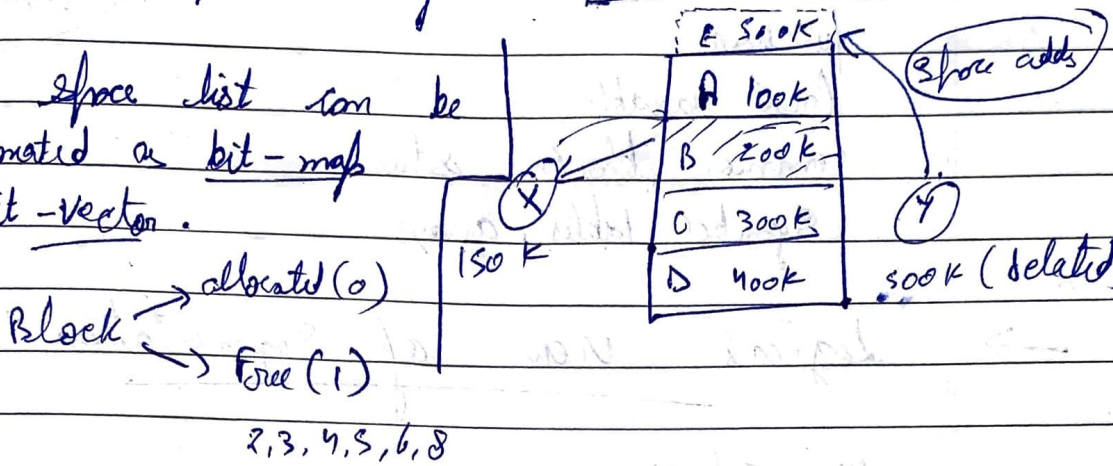
1) Pointer overhead
2) Multilevel Index

Page No.
Date :

- So, when file work is completed it free up the space. for that ↓

Free Space Management :- Free space list is used to keep track of all free disk blocks.

- Free space list can be implemented as bit-map or bit-vector.



Segmentation

Page No.
Date :

A program is a collection of segments

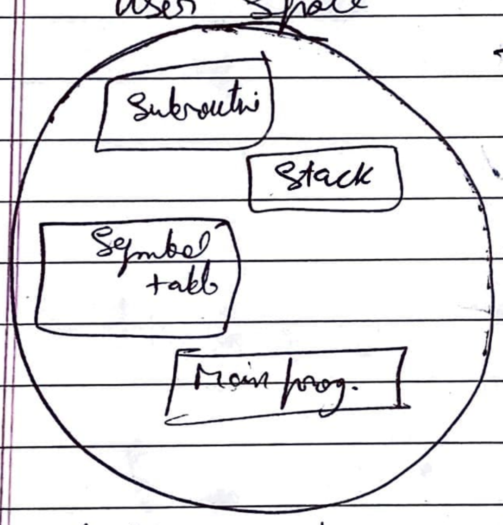
⇒ A Segment is a logical unit such as

- Main program
- Procedure
- function
- local variables
- Common blocks, stacks
- symbol tables, arrays

M/M mai jo sta data hai, vo ~~Virtual~~ physical memory

→ **Logical view of Segmentation :-**

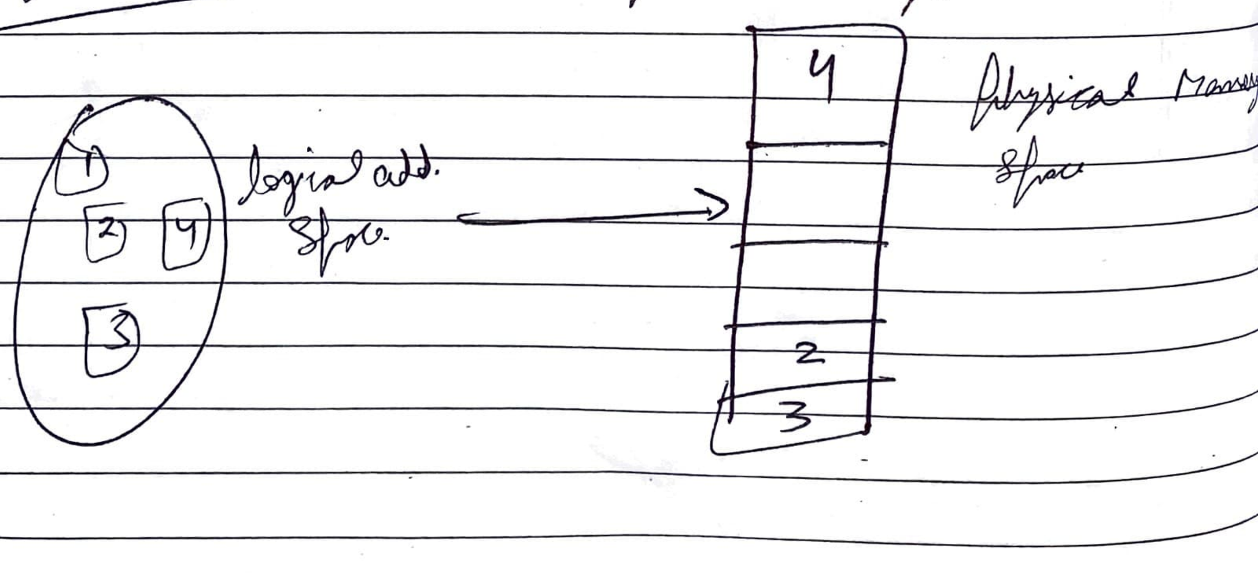
User Space



⇒ Specifies each address by 2 quantities

a) Segment name
b) Segment offset.

logical address ⇒ < Segment #, offset >



Segment Store in logical unit

Page No.
Date :

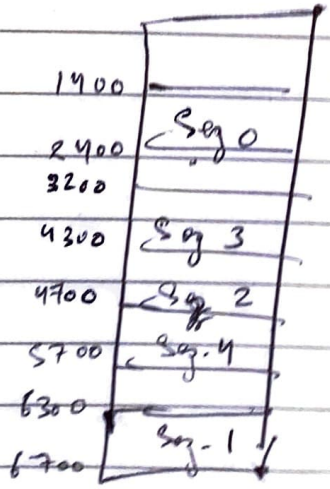
Segmentation Architecture :

- Logical add. contains 2 tuples
  < Seg. no., offset >
- Segment table — maps 2 dimensional phy. add :
  1) Base : Contains the starting Phy. add.
  2) limit : Specifies the link of Seg.
- Segment table base register (STBR) : [Base add. | del. leng]
  - Points to the seg. tables location in memory.
- Seg. table length register (STLR) :-
  - Indicates no. of seg used by program.

  Seg. no. is legal if S < STLR

Eg of Segmentation :-

	limit	Base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700



logical add space.