UNIT- V

File System Management :-

- A file is a collection of related inf" defined by its creator.
- In general file is a sequence of bits, bytes, lines or records. e.g. .txt [Personal details, account detail etc].

Some Example of files types -

→ Text File → Sequence of characters organized into lines

→ Source File → Sequence of subroutines or functions for specific tasks. [e.g. fn() calling fn()]

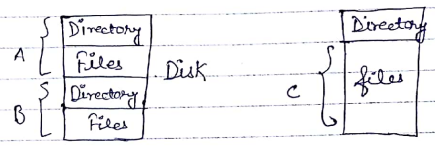
→ Object File → Collection of words, organized into loader record blocks.

File Attributes → Are the parameters used to keep track of file in O.S.

i)- Name
ii)- Identifier
iii)- Type
iv)- Location
v)- Size
vi)- Protection
vii)- Time & Date
viii)- User ID

File Operations → i)- Creating a file
ii)- Writing a file
iii)- Reading a file.
iv)- Repositioning a file
v)- Deleting a file
vi)- Truncation a file
↳ User may want to delete the content of a file.

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

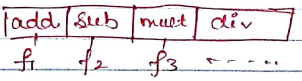


Symbol table that translates file name into their directory entries.

Operations on directory -

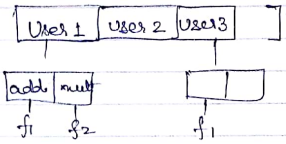
→ Search for a file
→ Create New file
→ Delete a file
→ List a directory
→ Rename a file
→ Traverse file system

Single-level Directory - All the files are contained in same directory.
Each file must have unique name.



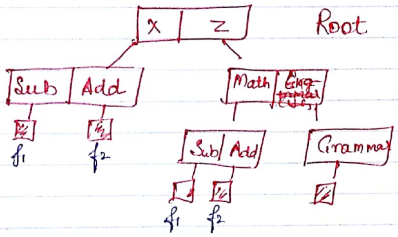
Reid & Taylor

(ii)- Two level Directory - Create a directory for each user -



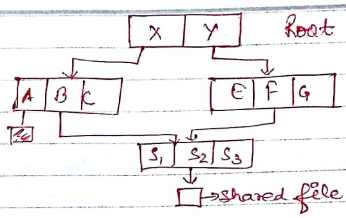
(iii)- Tree structure directories - Allows user to create their own sub-directories and organize their files accordingly.

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



(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]



File Allocation methods - Allocate space to the files so that disk space is utilized in an efficient manner.

Factors to consider -
- → Processing speed
- → Disk space utilization
- → Main m/m requirement

} OS should considered

Contiguous Allocation → Each file occupies a set of contiguous addresses on disk.

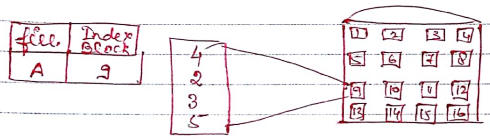
- → Linear Ordering
- → Location of a file is defined by the disk address of the first block and its length.
- → Both sequential and direct access is supported.
- → Easy to implement

Linked Allocation → Each file carries a list of links to disk blocks.

→ Directory contains link/pointer to first block of a file.
→ Effectively used in sequential access.
→ Inefficient in case of direct access file.

Indexed Allocation – Solves the problem of linked allocation.

In this all the pointers are brought together into one location called Index Block. Each file has its own index block.



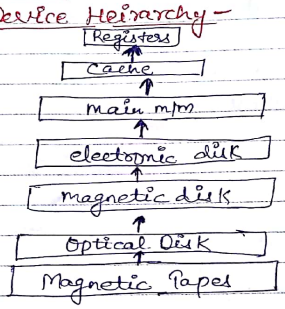
Direct access to block 2. First go to 9 and get the index block.

File Access Mechanism –
→ Sequential Access
→ Direct / Random Access
→ Indexed Sequential Access. → Indexed is searched sequentially and its pointer is used to access the file directly.

BOND WITH THE BEST

Disk Structure :–

Storage Device Hierarchy –



Magnetic disks provide the bulk of secondary storage for modern computer systems.
Each disk platter has a flat circular shape, like a CD, diameters from 1.0 to 5.25 inches.
The two surfaces of a platter are covered with a magnetic material.

★ A read-write head "flies" just above each surface of every platter.
★ The heads are attached to a disk-arm, which moves all the heads as unit.
★ The surface of a platter is logically divided into circular tracks, which are sub-divided into sectors.

Reid & Taylor

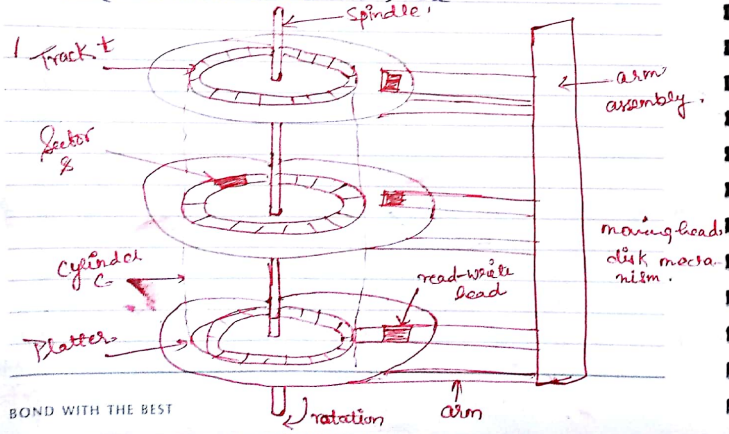
☆ When the disk is in use, a drive motor spins it at high speed. Most drives rotate 60 to 200 times per second.

☆ Sometimes the head will damage the magnetic surface. This accident is called a "head crash."

☆ A disk controller is built into each disk drive. and host controller.

☆ A disk drive is attached to a computer by a set of wires called an I/O Bus.

☆ Several kinds of buses are available, including, ATA, Serial ATA (advanced Technology attachment), USB (Universal serial Bus) etc.



BOND WITH THE BEST

Disk Scheduling:-

For the disk drives, meeting the responsibility entails having fast access time and large disk bandwidth.

Access time has two major components -

→ **Seek Time** – It is the time for the disk arm to move the heads to the cylinder containing the desired sector.

→ **Rotational Latency** – It is the additional time for the disk to rotate the desired sector to the disk head.

Disk Bandwidth → The disk BW is the total no. of bytes transferred, divided by the total time b/w the first request of service and the completion of the last transfer.

We can improve both the access time and the BW by scheduling the service requests in a good order.

☆ Whenever a process needs I/O to or from the disk, it issues a system call to the O.S. Then request specify some information like :-

Reid & Taylor

→ What m/m address for the transfer is
→ What the no. of sectors to be transferred is

★ If disk & host controller are available, then request is processed immediately.

★ If they are busy, then new requests will be placed in the queue of pending request for that drive.

★ In multiprogramming, many requests are in queue. Then which request will be chosen is done by disk scheduling algorithms.

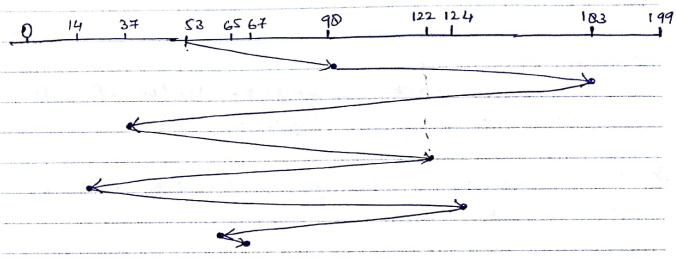
① FCFS Scheduling →

★ Simplest form of disk scheduling.
★ Intrinsically fair, but in general does not provide the fastest service.

Consider an example, a disk queue with requests for I/O to blocks on cylinders –

98, 183, 37, 122, 14, 124, 65, 67.

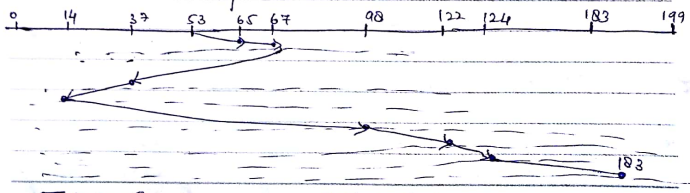
head starts = 53



Total movement of head ⇒ 640 cylinders.

SSTF scheduling :- (Shortest seek time First).

★ It selects the request with the minimum seek time from the current head position.
★ SSTF chooses the pending request closest to the current head position.



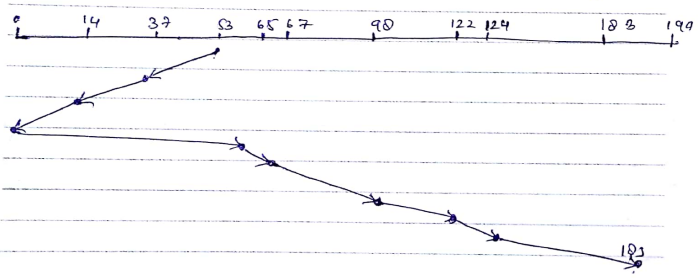
Total head move = 208 cylinders.

④- SCAN scheduling :-

★ Disk arm starts at one end of the disk and moves toward the other end, servicing requests as it reaches each cylinder, until it gets to the other end of the disk.

★ The head continuously scans back and forth across the disk.

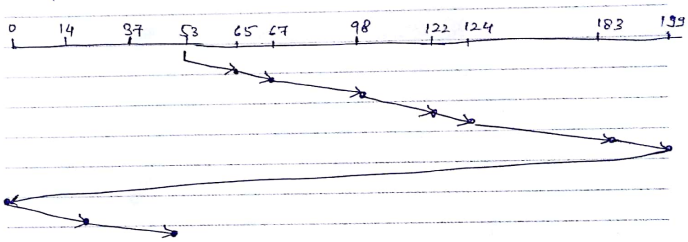
★ This is called sometimes "elevator algorithm."



⑤- C- SCAN scheduling —

★ C-SCAN moves the head from one end of the disk to the other, servicing requests along the way.

★ When the head reaches to other end of hard disk. however, it immediately returns to the beginning of the disk without servicing any request in return trip.



⑤- LOOK scheduling :-

★ Most commonly, the arm goes only as far as the final request in each direction.