UNIT 5

Mass-Storage Structure

1. Disk Structure :-

A. Magnetic disks provide bulk of secondary storage of modern computers.

→ Drives rotate at 60 to 200 times per second.

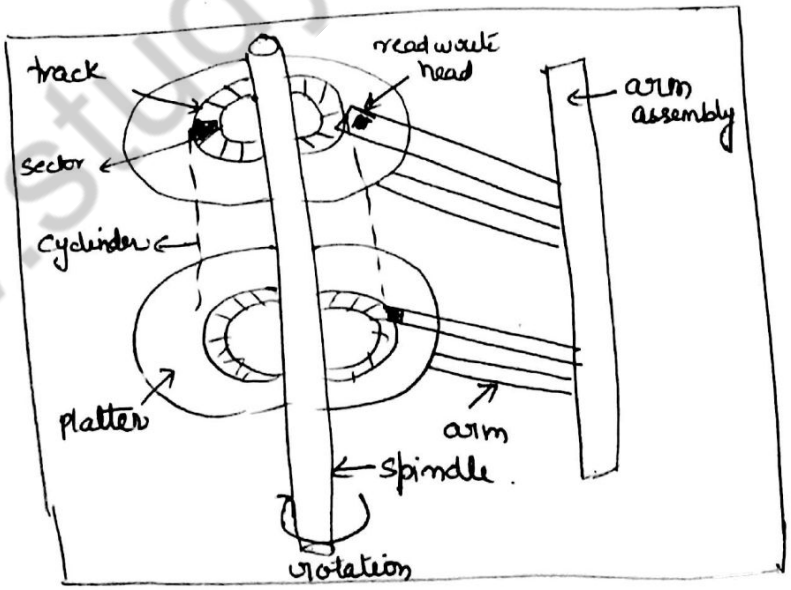
→ Transfer rate is a rate at which data flow between drive & computer

→ Positioning time (random-access-time) is time to move disk arm to desired cylinder (seek time) & time for desired sector to rotate under the disk head (rotational latency)

→ Head crash results from disk head making contact with the disk surface. That's bad.

B. Disk can be removable.

C. Drive attached to computer via I/O bus.



Disk Scheduling

A. The operating system is responsible for using h/w efficiently - for the disk drives, this means having a fast access time and disk bandwidth.

B. Access time has 2 major components :-

Seeking time :- is the time for the disk are to move the heads to the cylinder.

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

C. Minimize seek time.

D. Seek time ≅ seek distance

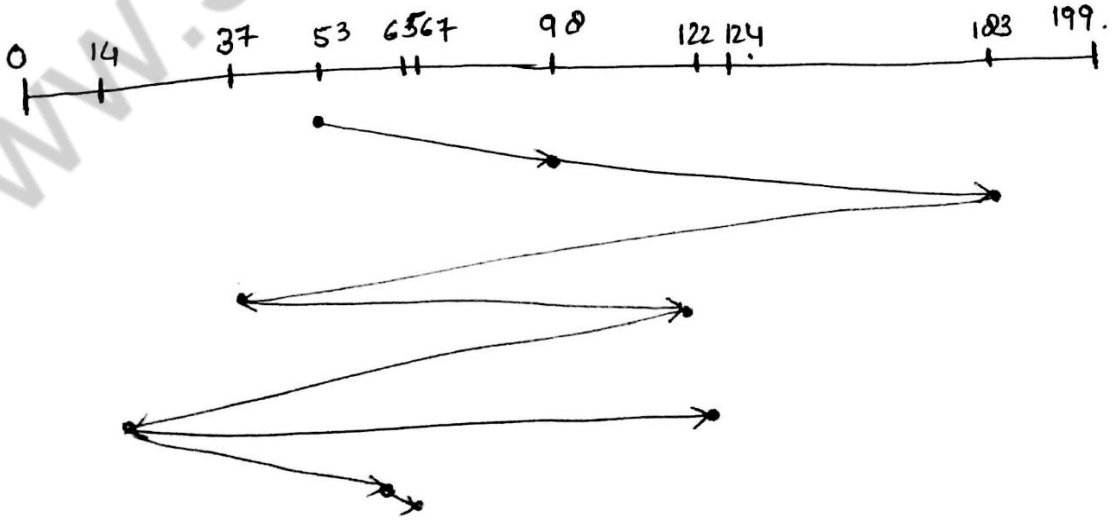
E. Disk bandwidth is the total number of bytes transferred, divided by the total time b/w the first request and the completion of last transfer.

⇒ Several algorithms exists to schedule the serving of disk I/O requests.

1. FCFS scheduling (First Come First Served)

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

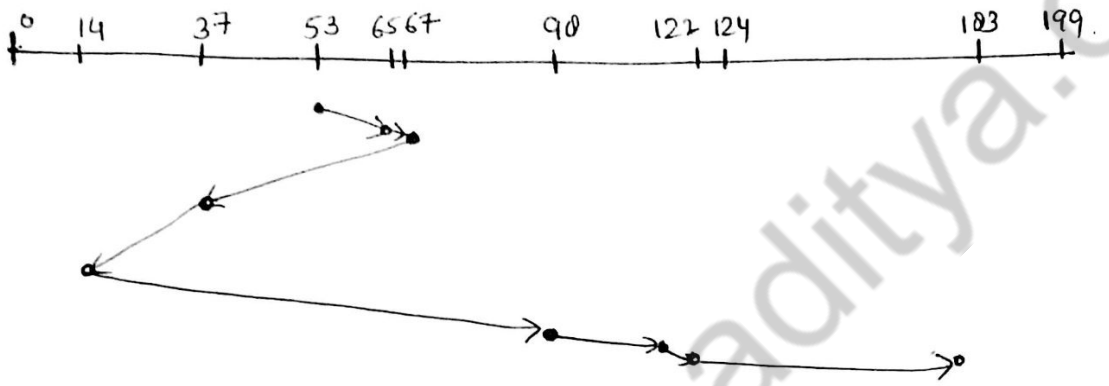
head starts at 53.



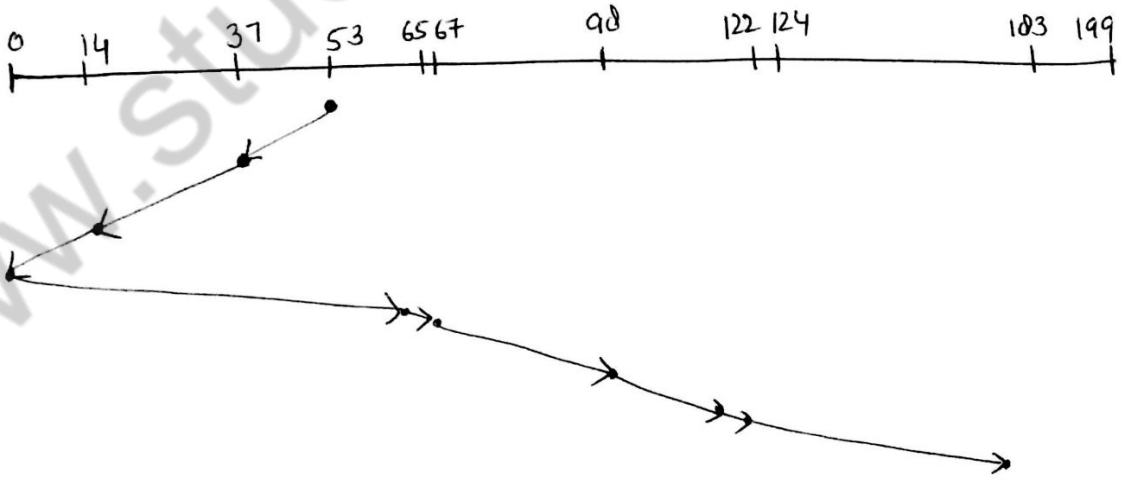
2. SSTF Scheduling (Shortest Seek time First).

- Selects the request with the minimum seek time from the current head position.
- SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests.

queue = 98, 183, 37, 122, 14, 124, 65, 65.
head starts at 53.



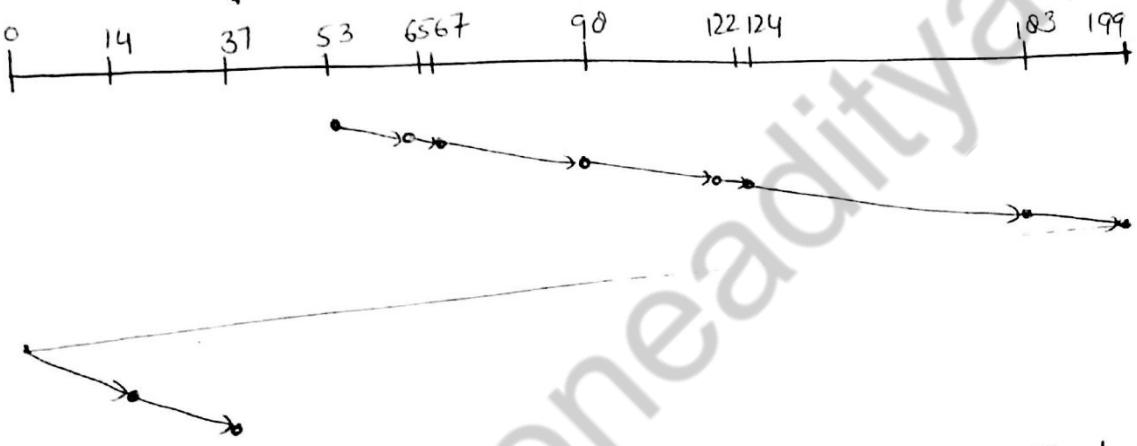
3. SCAN Scheduling:- The disk arm starts at one end of the disk, moves towards the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues. Sometimes called the elevator algorithm



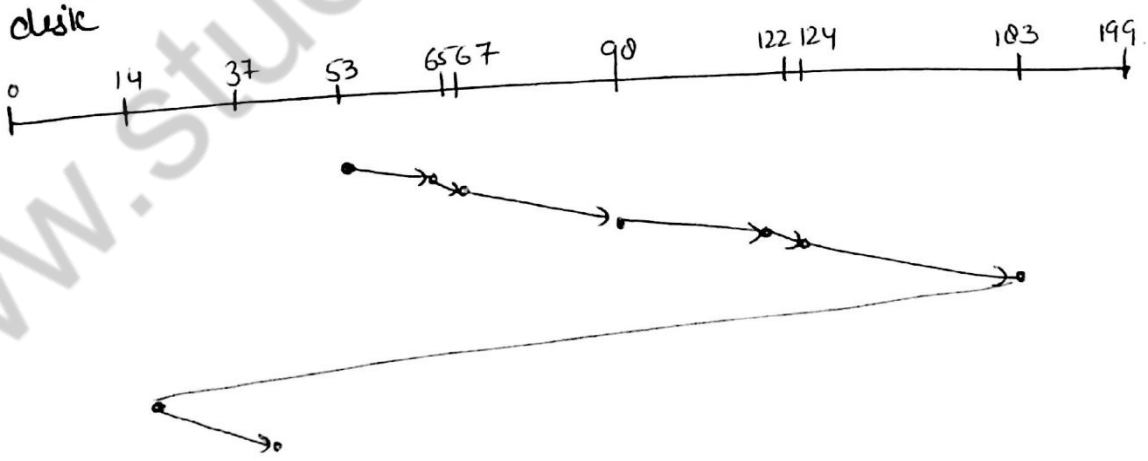
4. CSCAN Scheduling :- Provides a more uniform wait time than SCAN.

→ The head move from one end of the disk to the other, servicing requests as it goes. When it reaches the other end, however, it immediately returns to the begining of the disk, without servicing any requests on the return trip.

→ Treats the cylinders as a circular list that wraps around from the last cylinder to the first one.



5. C LOOK Scheduling :- Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk



Criteria for selecting a Disk Scheduling Algorithm

→ SSTF is common and has a natural appeal.
→ SCAN and C-SCAN performs better for systems that place a heavy load on the disk.
→ Performance depends on the number and types of requests.
→ Requests for disk service can be influenced by the file-allocation method.
→ The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary.
→ Either SSTF or LOOK is a reasonable choice for the default algorithm.

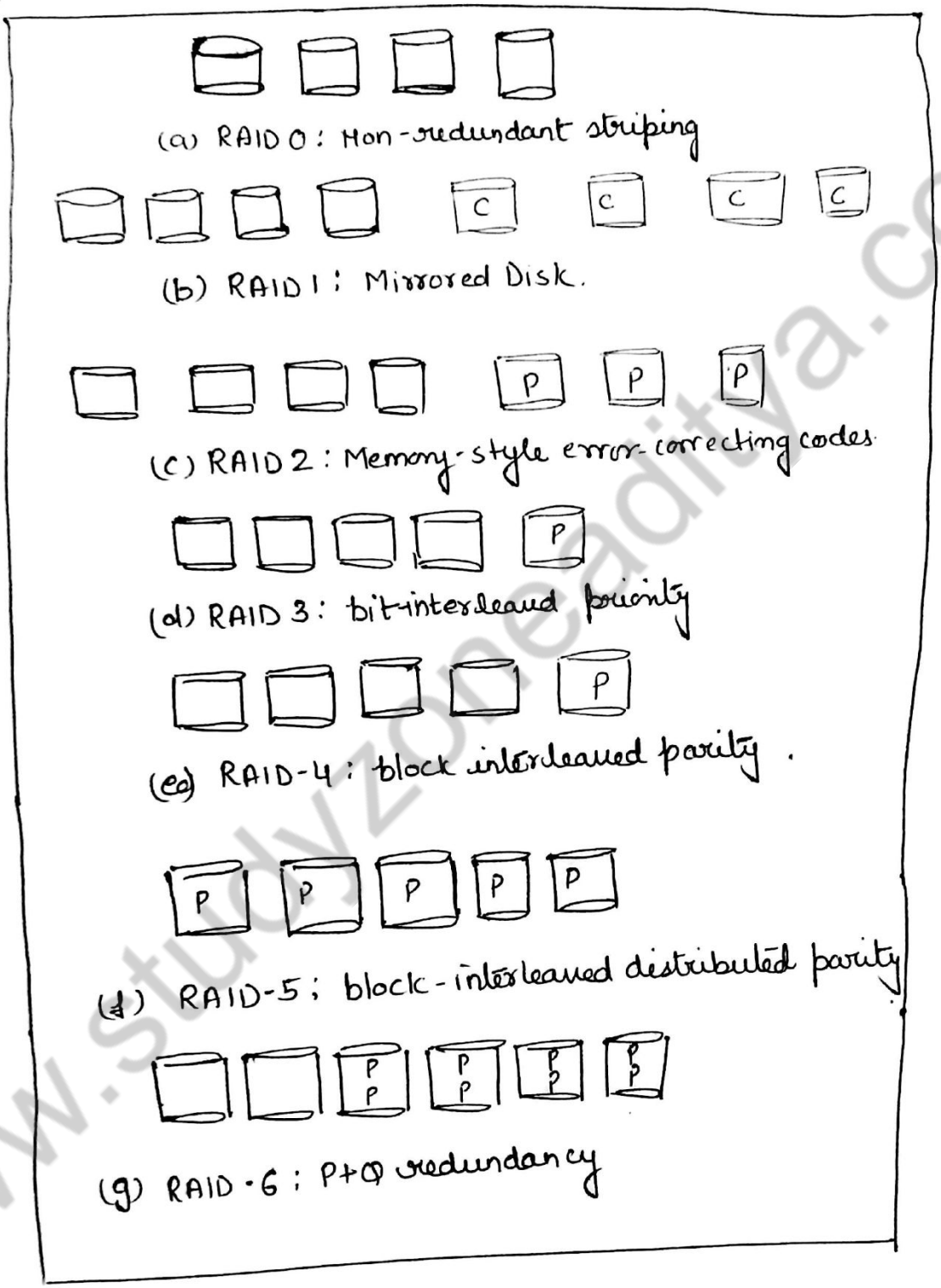
RAID Structure (Redundant Array Of Inexpensive Disk):

→ RAID - multiple disk drives provides reliability via redundancy.
→ Several improvements in disk-use technique involve the use of multiple disks working cooperatively.
→ Disk striping uses a group of disks as one storage unit.
→ RAID schemes improve the reliability of the storage system by storing redundant data.
  • Mirroring or shadowing keeps duplicate of each disk.
  • Block interleaved parity uses much less redundancy.
→ RAID is arranged into six different levels.

(i) **RAID Level 0:-** Level 0 refers to disk arrays with striping at the level of blocks but without any redundancy (such as mirroring or parity bits).

(ii) RAID Level 1 :-

Level 1 refers to disk mirroring. Fig (b) shows a mirrored organization.



(a) RAID 0 : Non-redundant striping

(b) RAID 1 : Mirrored Disk.

(c) RAID 2 : Memory-style error-correcting codes

(d) RAID 3 : bit-interleaved parity

(e) RAID-4 : block interleaved parity.

(f) RAID-5 ; block-interleaved distributed parity

(g) RAID-6 ; P+Q redundancy

(iii) RAID Level 2:- is also known as memory-style error-correcting code (ECC) organization. Memory systems have long detected certain errors by using parity bits. Each byte in a memory system may have a parity bit associated with it that records whether the no. of bits set to 1 in the byte set to 1 is even parity (parity = 0) or odd parity (parity = 1). If one of the bits in the byte is damaged (either a 1 becomes 0 or a 0 becomes 1), the parity of the byte changes and thus will not match the stored parity.

(iv) RAID Level 3:- or bit interleaved parity org^n, improves on level 2 by taking into account the fact that, unlike memory systems, disk controllers can detect whether a sector has been read correctly, so a single parity bit can be used for error correction as well as for detection.

(v) RAID Level 4:- Blocked interleaved parity org^n, uses block-level striping as in RAID 0, and in addition keeps a parity block on a separate disk for corresponding blocks from N other disks.

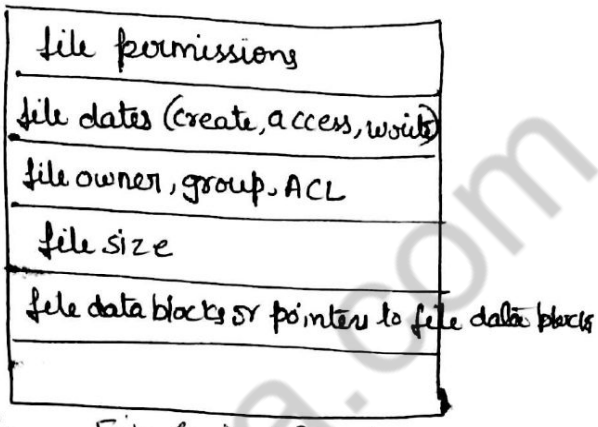
(v) RAID Level 5:- Block interleaved distributed parity differs from level 4 by spreading data and parity among all N+1 disks, rather than storing data in N disks and parity in one disk. For each block, one of the disks stored the parity & others stores data.

(vii) RAID Level 6: P+Q Redundancy scheme, is much like RAID Level 5 but stores extra redundant inf^n to guard against multiple disk failures. Instead of parity, error-correcting codes such as the Reed-Solomon codes are used. In the scheme shown in Fig. (g) 2 bits of redundant data are stored for every four bits of data - compared with 1 parity bit in Level 5 and the system can tolerate two disk failures.

File System

File Concept :- File organization

→ The operating system abstracts from the physical properties of its storage devices to define a logical storage unit, the file. Files are mapped by the operating system onto physical devices.



File Control Block

→ It contains data in type of numeric, character & binary etc.

→ It has file control block - storage structure consisting of inf^n about a file.

→ It has 2 types of record structure.

Simple record structure : Lines, fixed length, Variable length

Complex Structures :- Formatted Document, Relocatable load file

File Attributes :-

(i) Name : only information kept in human-readable form

(ii) Identifier : unique tag (number) identifies files within file system

(iii) Type : needed for systems that support different types.

(iv) Size : current file size.

(v) Location : pointer to file location on device.

(vi) Protection : controls who can do reading, writing, executing.

(vii) Time, date and user identification - data for protection, security & usage monitoring.

(viii) Inf^n about files are kept in the directory structure, which is maintained on the disk.

File Operations :-

(i) File is an abstract data type.
(ii) Create
(iii) Write
(iv) Read
(v) Reposition within file
(vi) Delete
(vii) Truncate
(viii) open(Fi) - search the directory structure on disk for entry Fi and move the content of entry to memory.
(ix) close(Fi) - move the content of entry Fi in memory to directory structure on disk.

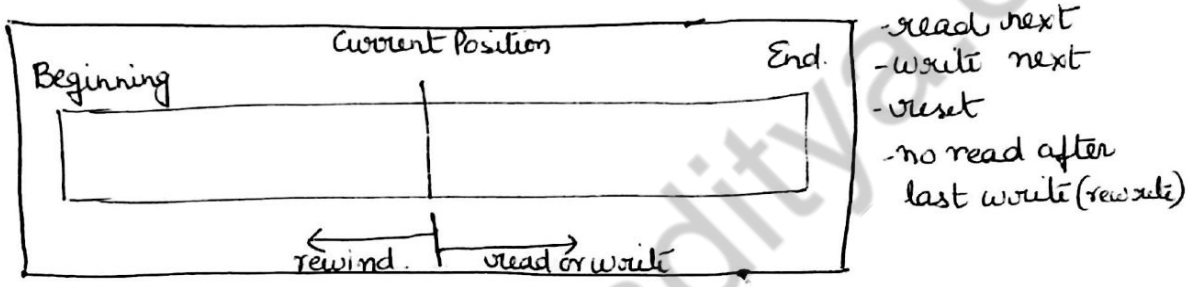
File Types

→ Executable
→ object
→ source code
→ batch
→ text
→ word processor
→ library
→ print or view
→ archive
→ multimedia

Access methods of a file

1. Sequential Access :-

The simplest access method is sequential access. Info in the file is processed in order, one record after the other. This mode of access is by far the most common; for example, editors and compilers usually access files in this fashion.



2. Direct Access :-

Another method is direct access (or relative access). A file is made up of fixed length logical records that allow programs to read and write records rapidly in no particular order. The direct-access method is based on a disk model of a file, since disks allow random access to any file block. For direct access, the file is viewed as a numbered sequence of blocks or records.

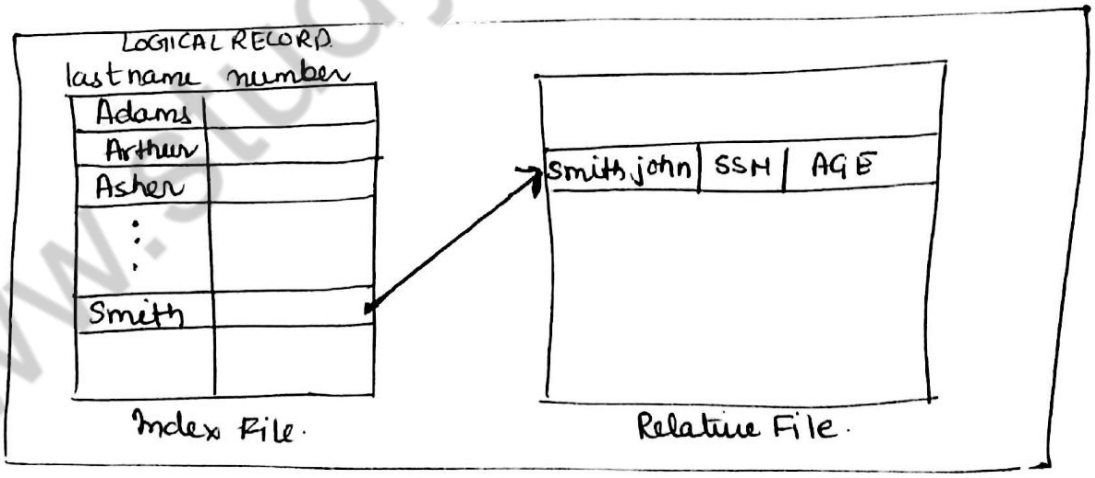
There are no restrictions on the order of reading or writing for a direct-access file.

Simulation of Sequential Access on a Direct Access file

Sequential Access	Implementation for direct access.
reset	cp = 0;
read next	read cp; cp = cp+1
write next	write cp; cp = cp+1;

3. Indexed Sequential Access Method (ISAM):-

Indexed sequential access method (ISAM) uses a small master index that points to disk blocks of a secondary index. The secondary index blocks point to the actual file blocks. The file is kept sorted on a defined key. To find a particular item, we first make a binary search of the master index, which provides the block number of the secondary index. This block is read in, and again a binary search is used to find the block containing the desired record.



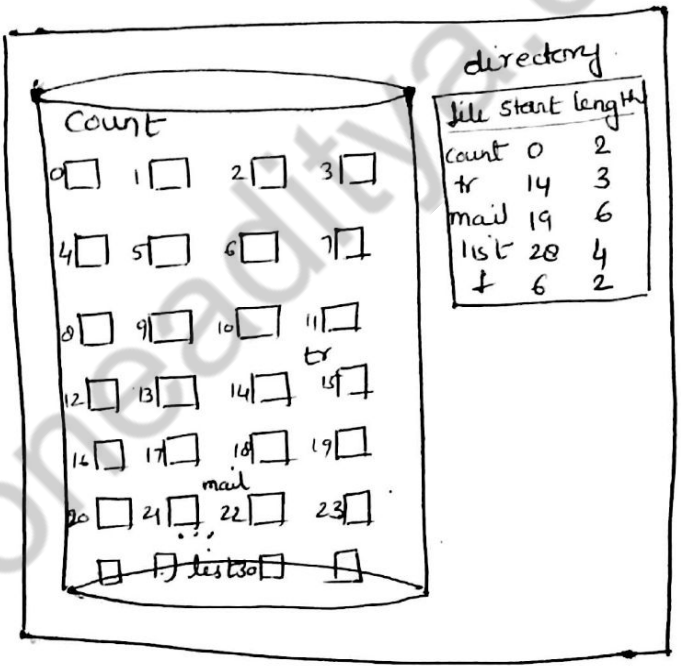
Allocation Methods

An allocation method refers to how disk blocks are allocated for files :-

(i) Contiguous allocation
(ii) Linked allocation
(iii) Indexed allocation

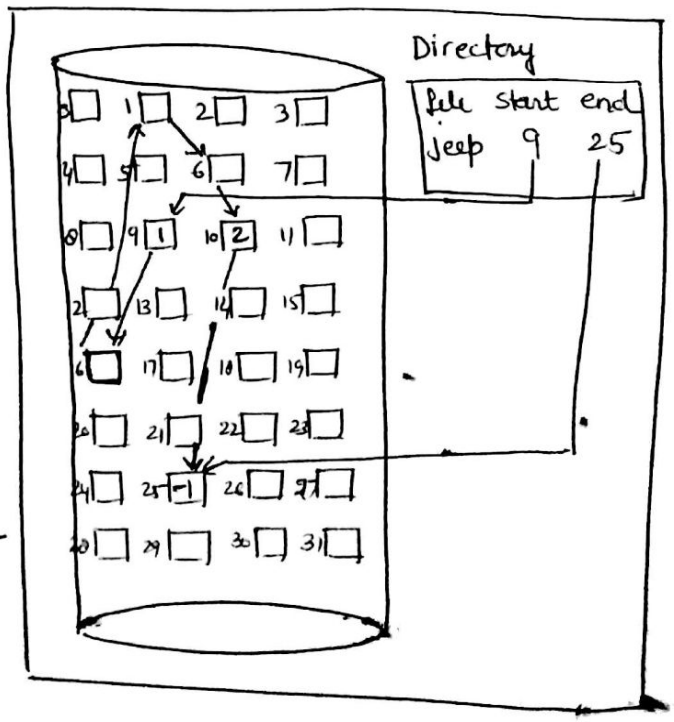
1. Contiguous allocation :-

- Each file occupies a set of contiguous blocks on the disk
- Simple - only starting location (block #) and length (no. of blocks) are required.
- Random access
- Wasteful of space (dynamic storage - allocation problem)
- Files cannot grow.



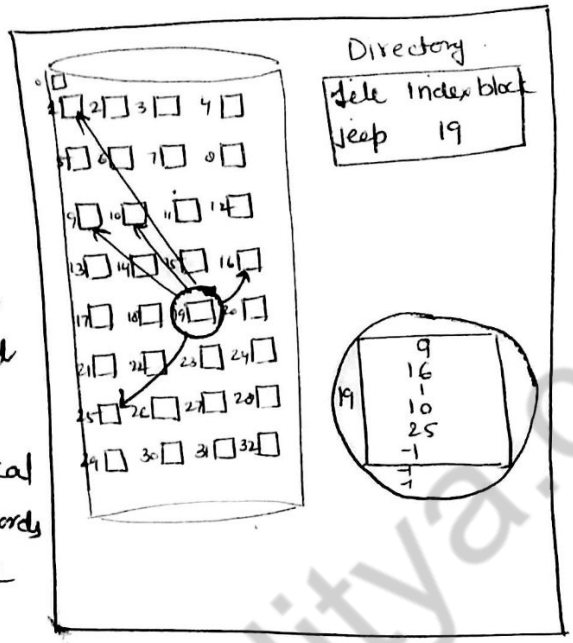
2. Linked Allocation :-

- Each file is a linked list of disk blocks : blocks may be scattered anywhere on the disk
- Simple - need only starting addr.
- Free-space mgmt system - no waste of space.
- No random access
- Mapping
  - File allocation table (FAT) - disk-space allocation used by MS-DOS and OS/2



3. Indexed Allocation

- Bring all pointers together into the indexed block.
- Need index table
- Random access
- Dynamic access without external fragmentation, but have overhead of index block.
- Mapping from logical to physical in a file of max size of 256K words and block size of 512 words. We need only 1 block for index table.



FILE SHARING

Multiple Users :-

→ User IDs identify users, allowing permissions and protections to be per-user

→ Group IDs allow user to be in groups, permitting group access rights.

Remote file system :-

→ Uses networking to allow file system access between systems.

- Manually via programs like FTP
- Automatically, seamlessly using distributed file system
- Semi automatically via the world wide web.

→ Client server model allows clients to mount remote file systems from servers.

- Server can serve multiple clients
- Client and user-on-client identification is insecure or complicated.
- NFS is standard UNIX client-server file sharing protocol.
- standard operating system file calls are translated into remote calls.

Failure Modes:-

→ Remote file systems add new failure modes, due to n/w failure, server failure.

→ Recovery from failure can involve state inf^n about status of each remote request.

→ Stateless protocols such as NFS include all inf^n in each request, allowing easy recovery but less security.

→ Client server model allows clients to mount remote file systems from servers.

- Server can serve multiple clients
- Client and user-on-client identification is insecure or complicated.
- NFS is standard UNIX client-server file sharing protocol.
- standard operating system file calls are translated into remote calls.

Failure Modes:-

→ Remote file systems add new failure modes, due to n/w failure, server failure.

→ Recovery from failure can involve state inf^n about status of each remote request.

→ Stateless protocols such as NFS include all inf^n in each request, allowing easy recovery but less security.

File Protection & Access Control

Protection mechanisms provide controlled access by limiting the types of file access that can be made (controlled access). Access is permitted or denied depending on several factors, one of which is the type of access requested. Several different types of operations may be controlled:

→ Read : Read from the file
→ Write : Write or rewrite the file
→ Execute : Load the file into the memory & execute it
→ Append : Write new info at the end of the file.
→ Delete : Delete the file and free its space for possible reuse.
→ List : List the name & attributes of the files.

(i) The most common approach to implement protect° is to make access dependent on the identity of the user.

(ii) The most general scheme is to implement identity dependent access is to associate with each file & directory an access-control list (ACL) specifying user names and the types of access allowed for each user.

(iii) The main problem with access list is their length. If we want to allow everyone to read a file, we must list all users with read access. This technique has two undesirable consequences:

(a) Constructing such a list may be a tedious task especially if we do not know in advance the list of users in the system.

(b) The directory entry, previously of fixed size, now needs to be of variable size, resulting in more complicated space mgmt.

(c) These problems can be resolved by use of a condensed version of the access list. To condense the length of the access control list, many systems recognize three classifications of users in connection ~~list~~ with each file.

→ Owner: The user who created the file is the owner.

→ Group: A set of users who are sharing the file and need similar access is a group, or work group.

→ Universe: All other users in the system constitute the universe.

Recovery :-

→ Consistency checking: compares data in directory structure with data blocks on disk, and tries to fix inconsistencie

→ Use system programs to back up data from disk to another storage device (floppy disk, magnetic tape, other magnetic disk, optical)

→ Recover lost file or disk by restoring data from backup.

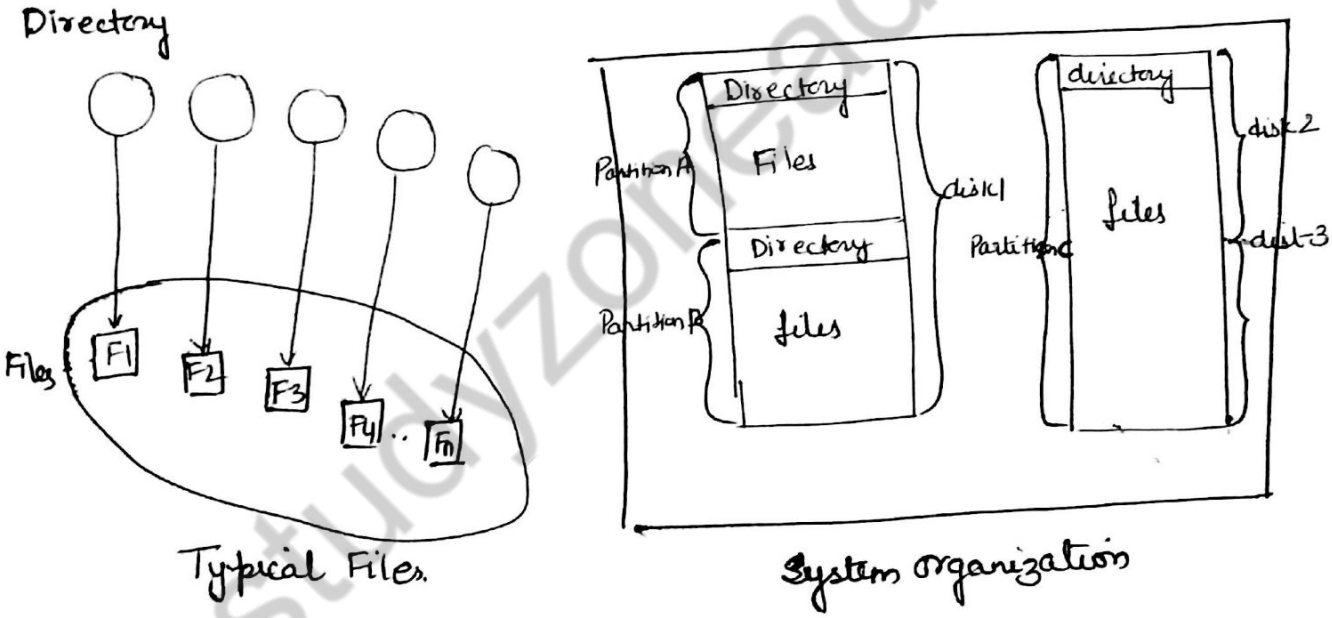
Directory Structure :-

→ To manage all the data we need to organize them. This org^n is usually done in 2 parts.

→ First, disks are split into one or more partitions, also as minidisks or volumes in the PC.

→ Second, each partition contains inf^n about files within it. This inf^n is kept in entries in a device directory or volume table of contents.

→ A collection of nodes containing inf^n about all files.

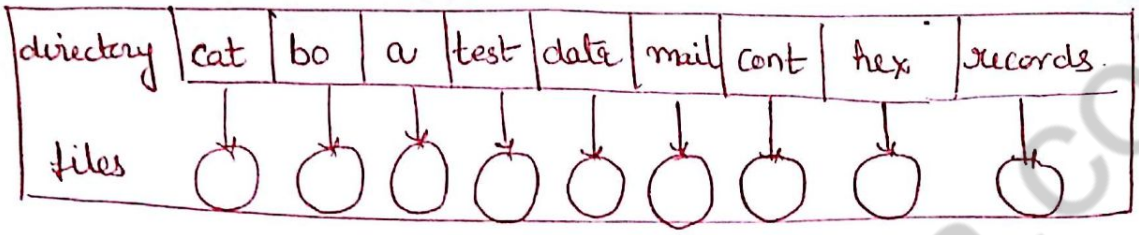


Single Level Directory :-

→ The simplest directory structure is the single level directory. All files are contained in the same directory, which is easy to support & understand.

→ A single level directory has significant limitations, since all files are in the same directory, they must have unique names.

→ It is difficult to remember the names of all the files as the number of files increases.



Two level Directory