File - System in OS :

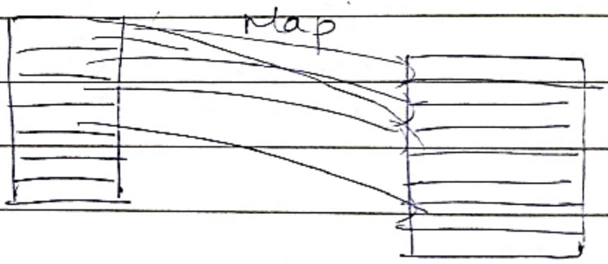
~~Softw.~~

File : It is a collection of related information that is recorded on secondary storage

file-system is a software which manages the storage & retrieval of files.

File system stores the mappings of the blocks to various sectors.

User → file → folder/directory → file system



Operations on files	File attributes
1) Creating	1) Name ~~6) Modified date~~
2) Reading	2) Extension (.jpg, .docx etc.)
3) Writing	3) Identifier (file ID for OS)
4) Deleting	4) Location
5) Truncating	5) Size
6) Re-positioning	6) Modified date, created date
	7) Protection / Permission (read/writ
	8) Encryption, Compression

Attributes are meta-data

Deletion: Entire file gets deleted along with attributes

Truncation: Contents of file are deleted but attributes remain as it is

Re-positioning: When a file is opened initially the R/W head is at the beginning of the file, if we change the position of pointer we do its re-positioning.

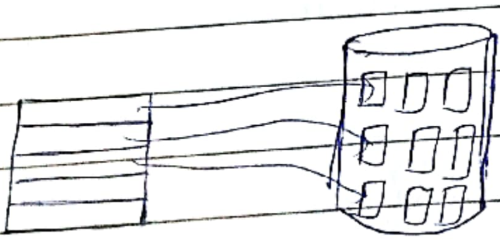
For e.g. $\underset{\uparrow}{a}bcd\underset{\uparrow}{1}234$

If we change the location of pointer from a to 1, then we re-position it.

Allocation Methods:

$$\begin{tikzcd} & \text{} \arrow[dl] \arrow[dr] & \\ \text{Contiguous allocation} & & \text{Non-contiguous allocation} \arrow[d] \arrow[dd, bend left] \\ & & \text{Linked list allocation} \\ & & \text{Indexed allocation} \end{tikzcd}$$

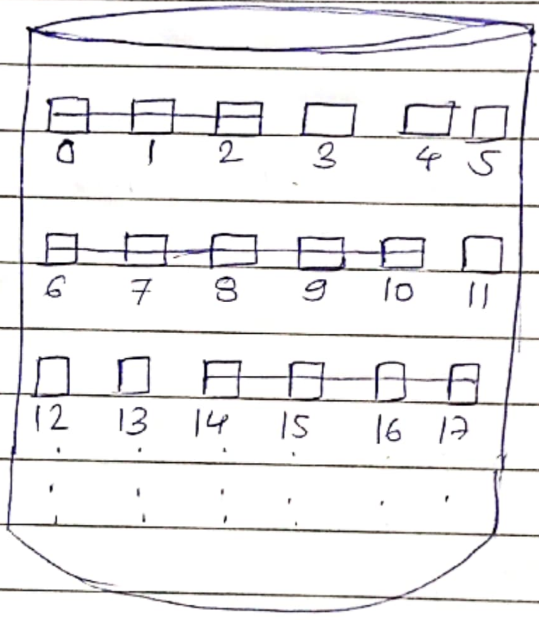
File is divided ~~into~~ logically & stored in the sectors of the disk on HD.



Purpose:
1) Effecient disk utilization
2) Faster access.

Contigouous allocation:

A single continuous set of ~~to~~ blocks is allocated to a file at the time of file creation



Directory

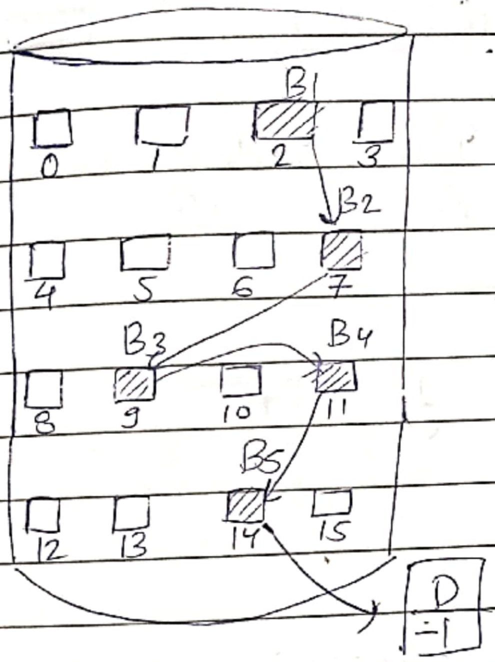
file	start	length
A	0	3
B	6	5
C	14	4

Advantages:
→ Multiple blocks can be read in at a time to improve I/O performance for sequential processing.
→ Easy to retrieve a single block.
e.g. if a file starts at block b & ith block is wanted then its location is b + i − 1
→ Easy to implement
→ Excellent read performance

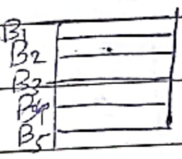
Disadvantages:
→ Internal fragmentation is there.
→ External — " ———— would be there, making it difficult to find contiguous blocks of space of

suffecient length.
→ We need to declare file size at the time of creation
→ Difficult to grow file.

Linked List Allocation: (NON-CONT.)



file	start
F1	2



Data	Pointer

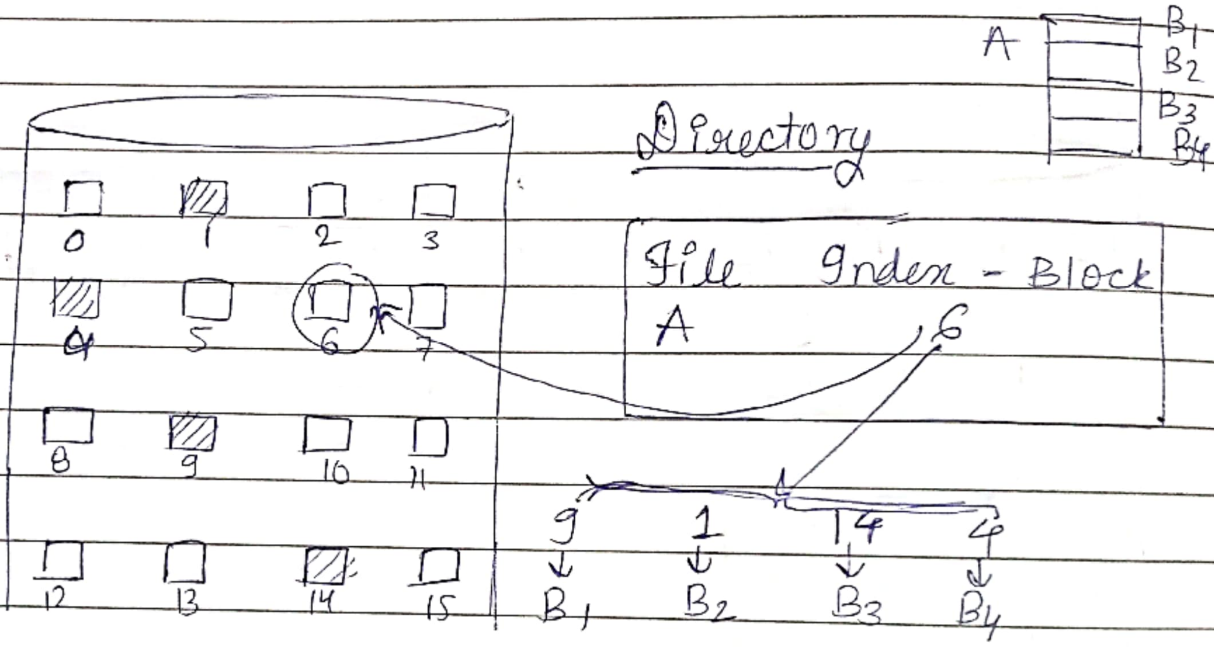
~~Alo~~ Allocation is on a individual block basis. Each block contains a pointer to the next block in the chain. File table just needs a single entry showing the starting block & length of each file

Advantages:
→ No external fragmentation → (If there are non-contiguous empty blocks, they can be filled here)
→ File size can increase

Disadvantages:
→ Internal fragmentation exists in last disk block of a file.

→ Overhead of maintaining pointer in every disk block.
→ If pointer of any disk block is lost, the file would be truncated.
→ Supports ~~only~~ only sequential access of files.

~~I~~ Indexed Allocation:



In this case, the file allocation table ~~is~~ contains a seperate one-level index for each file. The index has one entry for each block allocated to the file.
(This concept is same like a the index of a book, like every book has its index which indicates the posn of a particular chapter, each file has a index)

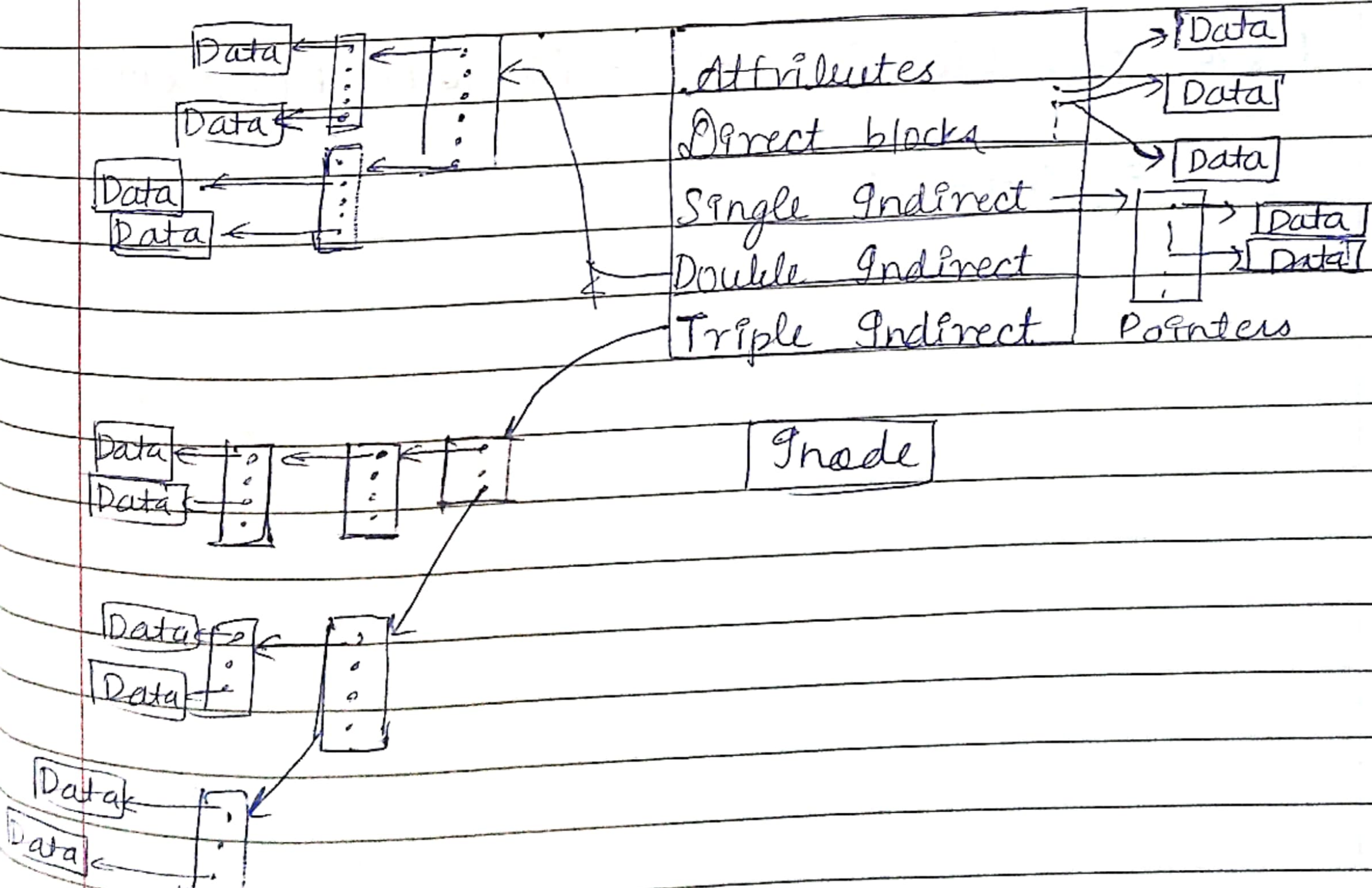
Advantages:
→ Supports direct as well as sequential access
→ No external fragmentation

Disadvantages

→ Pointer overhead
→ Multi-level index (if the file is too big, ∴ index can't fit in a block, so we have to maintain ~~multi-level~~ multi-level index)

UNIX file system uses I-node

UNIX INODE STRUCTURE



Q A file system uses Unix Inode data structure which contains 8 direct block addresses, 1 indirect block address, 1 double & 1 triple indirect block. The size of each disk block is 128 B & size of each block address is 8 B. Find max. possible file size.

Direct block addr = 8

One indirect block addr. = $\frac{128\,B}{8\,B} = \frac{2^{7}}{2^{3}} = 2^{4} = 16$

Double indirect = 16×16

Triple —"— = $16 \times 16 \times 16$

$\therefore$ Total = $8 + 16 + 16^{2} + 16^{3}$ pointers

Max. possible file size = $(8 + 16 + 16^{2} + 16^{3}) \times 128\,B$

= 547 KB