Operating system

Unit - 5.

File Management

* **Introduction** :- File management is one of imp. functions of an operating system. It manages files and directories i.e. folders stored on secondary storage devices such as hard disks, SSDs, pendrives etc.

File:-

A file is a collection of related information stored on a storage device with a specific name.

e.g. student.txt, program.c, result.xlsx.

File management:-

File management is the process by which the operating system creates, stores, organizes, accesses, modifies and deletes files and directories.

↳ main functions of filemanagement includes:
File creation, deletion, Reading, writing, modification, renameing, directory management, file protection, organization, storage management etc.

* **File attributes** :-

File attributes are properties or characteristics of a file maintained by operating system.

↳ They provide information about file & helps OS to manage and protect it.

common file attributes :-

1) Name — The name used to identify file
e.g. student.doc

2) Identifier — A unique number assigned to file by OS.

3) Type — specifies the type of file, such as text, image, audio, executable etc.

4) Location — Indicates location of the file on the storage device.

5) size — specifies amount of storage space occupied by file.

6) Protection — specifies who can read, write & execute the file.

7) Time & date — stores information about when the file was created, modified or accessed.

8) owner / UserId — Identifies the user who owns or created the file.

e.g. student.txt file has:

Name - student.txt	Protection - Read or write
TYPE :- Text file	created - 10 Aug 2026
Size - 15 KB	owner - student
Location - Hard disk	

* File operations :-

File operations are actions that the OS performs on files. The OS provides, various operations to allow users & programs to create, access, modify and manage files.

* common file operations are:

(1) create :-
It is used to create new file.

(2) open :-
The open operation makes a file available for reading or writing.

(3) Read :-
The read operation retrieves data from a file.

(4) Write :-
The write operation stores data in a file.

(5) Append :-
The append operation adds new data at the end of existing file without removing previous content.

(6) seek :-
The seek operation moves the file pointer to a particular position in the file.
e.g. moving directly to the 100th byte of a file to read from that location.

(7) close :-
The close operation closes a file after required work is completed.

⑧ Delete:- The delete operation removes the file from storage system.

⑨ Truncate:-

The truncate operation removes the contents of a file while keeping the file available.

⑩ Rename:-

The rename operation changes a name of the file.

* File Types:-

A file type tells the operating system and user what kind of information is stored in a file.

↳ The file type is commonly identified using the file extension.

e.g. common file types:

① Text file	.txt	notes.txt	stores text information.
② Source code file	.cpp, .java .py	program.cpp	stores program source code.
③ Executable file	.exe	setup.exe	stores executable programs.
④ Document file	.doc .pdf	report.pdf	stores documents
⑤ Image file	.jpeg, .png	photo.jpg	stores images
⑥ Audio file	.mp3	song.mp3	stores sound
⑦ Video file	.mp4	movie.mp4	stores video

* File system structure:-

A file system is the method used by an operating system to organize, store, name, manage files and directories on a storage device.

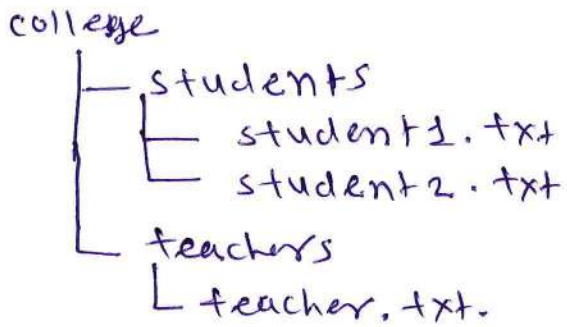
* main components of file system structure-

(1) Files:-

It contains actual user data or program information.

(2) Directories:-

A directory is used to organize and store information about files & other directories.



(3) Subdirectories:-

A directory inside another directory is called a subdirectory. e.g. college → students.

(4) Root directory:-

The root directory is top level directory from which other directories and files are organized.

Where / is root directory.

Types of file system structure :-

Operating system can organize files using different directory structures.

1. single-level directory

All filer are stored in a single directory.

```
directory
  ├── file1.txt
  └── file2.txt.
```

Advantage – Very simple to understand.

disadvantage – It becomes difficult to manage files when the number of filer increases.

2. Two-level Directory

Each user has a separate directory.

```
Root
  ├── User1
  │     ├── file1.txt
  │     └── file2.txt
  └── User2
        ├── file1.txt
        └── file2.txt
```

Different users can have filer with same name.

3. Tree-structured Directory :-

Directories can contain files as well as subdirectories.

```
Root
  ├── college
  │     ├── students
  │     │     ├── student1.txt
  │     │     └── student2.txt
  │     └── Teachers
  │           └── teacher1.txt
  └── Documents
        └── Reports.pdf
```

This is commonly used, because it provides a clear & organized structure.

* File Accessing methods:-

File accessing methods define how data is read from or written to a file. The operating system provides different methods for accessing data stored in files.

The two common methods are:

1. Sequential Access.
2. Direct Access.

(1) Sequential Access:-

In sequential access data is accessed one record after another in fixed order from beginning of the file.

↳ It is similar to reading a book page by page.

e.g.

Suppose a file contains student records:

Record 1 → Ajay
Record 2 → Ravi
Record 3 → snehal

To access record 3 the sequential access method uses following path.

Record 1 → Record 2 → Record 3

* characteristics:-

1. Data is accessed in fixed sequence.
2. Records are processed from beginning to end.
3. It is simple to implement.
4. It is suitable when all records need to be processed.
5. Accessing last record may take more time.

Applications :-

① Text files ② Log files ③ Audio/video streaming.
④ Backup files etc.

② Direct Access method :-

Direct access also called as random access, allows a program to access a particular location or record directly without reading all previous records.

↳ It is similar to open a book directly at page No. 50 instead of reading pages from 1 to 49.

e.g. suppose a file contains:

Record 1 → Ajay
Record 2 → Ravi
Record 3 → snehal

To access record 3, the direct access method directy move to Record 3

Record 4 ← Direct access.

*characteristic :-

① Data can be accessed in any order.
② A particular record can be accessed directly.
③ It is faster when specific records need to be accessed frequently.
④ It is useful for large files.
⑤ The system uses a file position or record number to locate data.

*Applications :-

① Databases ② Disk files ③ Banking systems.
④ Student record systems. ⑤ Employer database.

5.3 File Allocation methods:

File allocation is the method used by an OS to determine how the blocks of a file are stored on secondary storage such as hard disk or SSD.

A file is divided into one or more blocks, and these blocks must be allocated to the file in such a way that the file can be accessed efficiently.

The major file allocation methods are:

1. Contiguous Allocation
2. Linked Allocation
3. Indexed Allocation

① Contiguous Allocation:

In contiguous allocation, all blocks belonging to a file are stored in consecutive (adjacent) disk blocks.

e.g: If a file requires 5 blocks and block 21 is the starting block, the file may occupy:

$21 \longrightarrow 22 \longrightarrow 23 \longrightarrow 24 \longrightarrow 25$

The directory needs to store mainly:

- Starting block number
- Length / number of blocks.

e.g: Suppose a file File1 requires 4 blocks and is allocated starting from block 10.

The directory entry can contain:

File Name	Starting Block	Length
File1	10	4

Therefore, the OS knows that the file occupies blocks: 10, 11, 12 and 13.

Advantages of Contiguous Allocation:

- Simple to implement
- very fast sequential access.

- very fast direct / random access.
- less disk head movement.
- Good performance.

Disadvantages of Contiguous Allocation:

- causes external fragmentation.
- Difficult to increase the size of a file if adjacent blocks are occupied.
- finding a sufficiently large contignous space can be dificult.

Example: CD / DVD - style storage and some systems that know the file size in advance can use contiguous allocation.

(2) Linked Allocation:

In linked allocation, the blocks of a file can be stored anywhere on the disk. Each block contains a pointer to the next block of the same file. The blocks can be located at completely different positions on thedisk.

e.g: Suppose file2 requires five blocks:

7 ⟶ 19 ⟶ 4 ⟶ 25 ⟶ 12 ⟶ NULL

The directory stores the address of the first block.

File Name	Starting Block
File2	7

Each block contain a pointer to the next block

Block	Next
7	19

⟶

Block	Next
19	4

⟶

Block	Next
4	25

⟶

Block	Next
25	12

⟶

Block	Next
12	NULL

How Linked allocation works:

When a file is created, the OS allocates any available disk block. Suppose the first block is block 7. The next available block may be block 19. The first block therefore contains a pointer to 19.

The last block contains a special value such as NULL to indicate the end of the file.

Advantages:

- No external fragmentation.
- File can grow easily.
- Efficient use of free space.
- File size need not be known in advance.

Disadvantages:

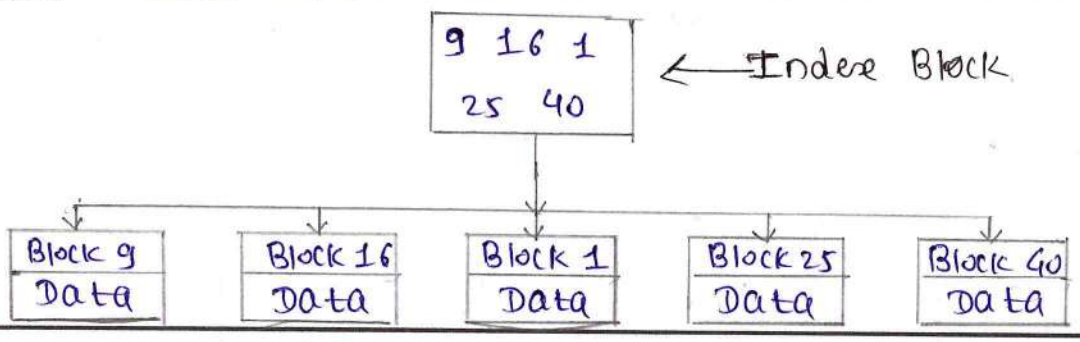
- Direct / random access is slow because blocks must be followed sequentially.
- pointer space is required in each block.
- If a pointer is damaged, the remaining part of the file may be become inaccessible.
- More disk I/O may be required.

Example: FAT (File Allocation Table) is a well-known approach related to linked allocation, where the pointers are maintained in a seperate table.

③ Indexed Allocation:

Indexed allocation solves many of the problems of linked allocation by maintaining an index block for each file. The index block contains the addresses of all disk blocks belonging to the file. The actual data blocks can be located anywhere on the disk.

e.g: Suppose a file requires five blocks:
An index block contains their addresses:



Advantages:

- No external fragmentation
- Supports direct / random access.
- Files can grow easily.
- Data blocks can be located anywhere on the disk.

Disadvantages:

- Extra space is required for the index block.
- Large files may require multiple levels of indexing.
- Small files may waste space because the index block.

Example:

UNIX / Linux inode-based file systems use an indexed approach, including direct and indirect block pointers.

5.4 Directory Structure:

A directory is a special structure maintained by the OS that is used to organize, store and manage information about files and other directories.

A directory generally contains information such as:

- File name
- File type
- File Location
- File size
- File permissions
- Date and time of creation / modification
- File owner
- other file attributes.

The organization of directories is called directory structure.

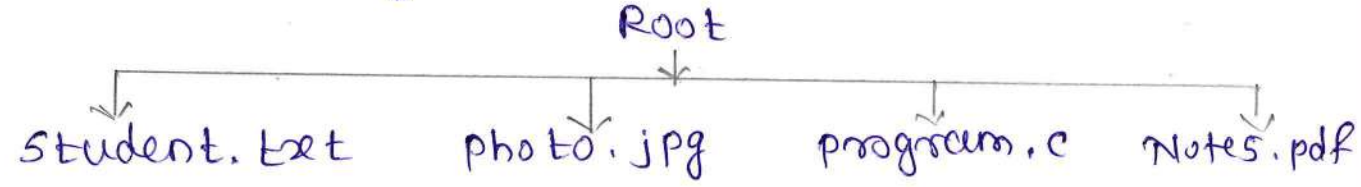
The major types of directory structure are:

① Sigle - Level Directory
② Two - Level Directory
③ Tree - Level Directory.

① Single - Level Directory :

A single-level directory structure, there is only one directory for the entire system. All users store their files in same directory. All files are stored at the same directory level.

Example: Suppose three users create files. All these files are stored in the same directory "Root".



Advantages :

- simple structure
- Easy file management
- Low overhead
- Easy implementation

Disadvantages :

- File name conflicts
- Not suitable for multiple users
- Difficult to organize files
- Searching becomes dificult.

Applications:

Single-Level directories are suitable for:

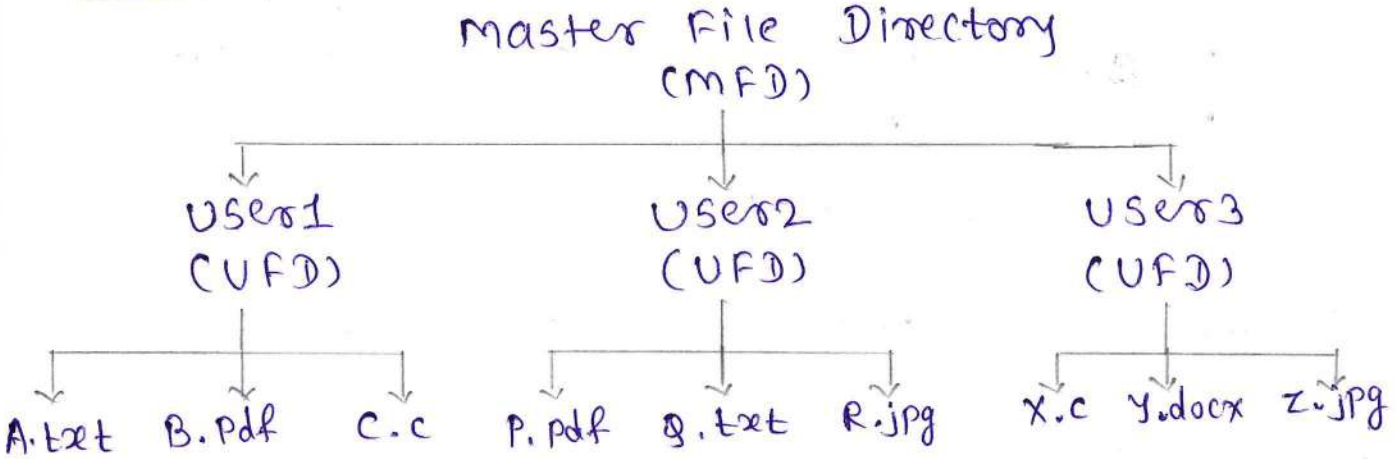
- Small systems
- Simple applications
- Systems with a small number of files.
- Basic or early operating systems.

② Two-Level Directory:

The two-level directory structure was developed to overcomes some of the problems of the single-level directory.

In this structure, there is a Master File Directory (MFD) and a seperate User File Directory (UFD) for each user.

Structure:



The master File Directory (MFD) contains information about users.

Each user has a seperate User file Directory (UFD) containing that user's files.

Advantages:

- Avoids name conflicts between users
- Better organization
- provides User-level seperation

- Easy file searching
- Better than ~~sigle~~ - single-level directory.

Disadvantages:

- Limited grouping
- Sharing files can be dificult
- Not suitable for complese systems.

Applications:

Two-level directories are useful in:

- multi-user operating systems.
- systems where users need seperate file spaces.
- simple networked or shared systems.

③ Tree-Structure Directory

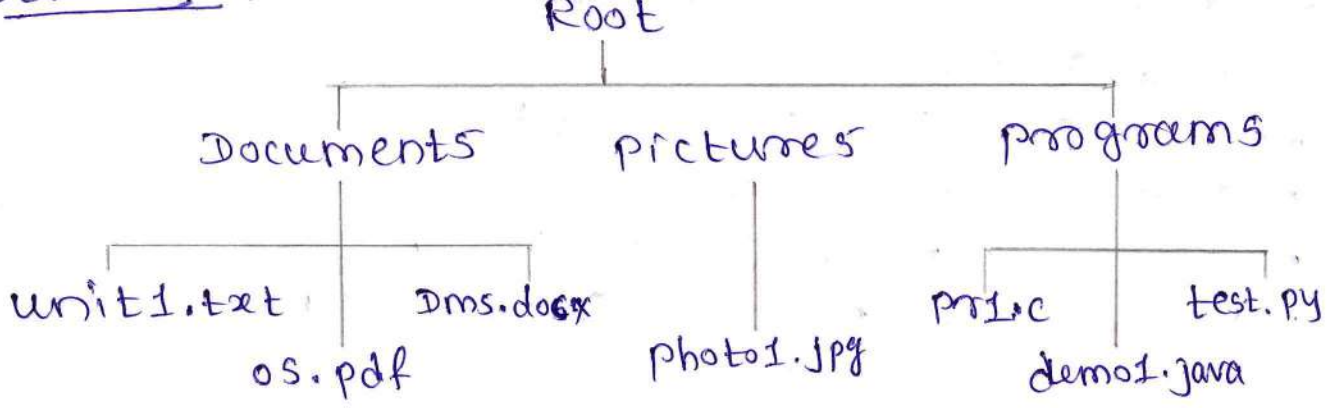
The tree-structure directory is one of the most commonly used directory structures in modern operating systems.

In this structure, directories can contain:

- files
- subdirectories
- other subdirectories

The structure resembles a tree, with a root directory at the top.

Structure:



Types of paths in Tree-structure Directory:

A major advantage of tree-structure directories is use of path names. A path specifies the location of a file or directory.

There are two common types of paths:

① Absolute path
② Relative path

① Absolute path:

An absolute path specifies the complete path from the root directory to the file.

e.g: /root/students/TYCW/os.txt

The path starts from the root.

② Relative path:

A relative path specifies the location of a file relative to the current directory.

e.g: If the current directory is:

/root/students/TYCW

then: os.txt can be used to access the file.

Advantages:

- Excellent organization
- Avoid naming conflicts
- Supports user organization
- Better file management
- Supports large numbers of files.
- Easy file searching
- Supports subdirectories
- Supports access control.

Disadvantages:

- more complex implementation
- path management
- Directory management overhead.
- File sharing requires additional mechanisms.