# Operating Systems (UNIT-5)

# Unit-5

**File Systems**

File Naming, File Structure, File Types, File Access, File Attributes, File Operations, An example program using File-System calls, File-System Layout, Implementing Files

# File Naming

A file is a named collection of related information that is recorded on secondary storage.

Contiguous logical address space

Types:

- Data
  - numeric
  - character
  - binary
- Program

Contents defined by file's creator

- Many types
  - Consider **text file, source file, executable file**

# File Naming

Files are an abstract mechanism, the most important characteristic of any abstract mechanism is the way that the objects being managed are named.

The rule for file naming varies from system to system, but all the current OSs allow strings as file names. Therefore, myfile, file1, my_file, etc. are the three legal file names.

Digits and special characters can also be used to name a file, such as myfile, file-2-1, file_21, 32, etc., which are also legal file names.

The naming of files is supported by many file systems as long as the names contain all 255 possible characters.

Almost every OS supports two-part file names. These two parts of the file names are separated by a period or dot (.).

The part that comes after the period (.) is known as the file extension, and it typically provides some kind of information regarding the file

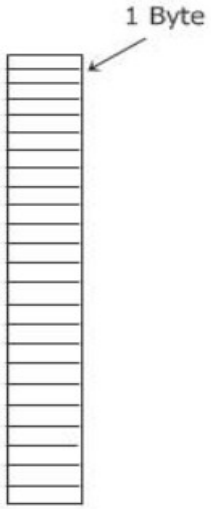
# File Naming

| File Extension | File Meaning |
|---|---|
| myfile.bak | This indicates the existence of a backup file. |
| myfile.c | This denotes a source file for the C programming language. |
| myfile.gif | This denotes an image file in GIF format. |
| myfile.hlp | This indicates the presence of a help file. |
| myfile.html | This is an HTML (HyperText Markup Language) file. |
| myfile.jpg | This denotes an image file in JPG format. |
| myfile.mp3 | This denotes an MP3 music or audio file in which the music has been encoded in MPEG Layer 3 audio format. |
| myfile.mpg | This is an MPEG video file containing a movie encoded with the MPEG standard. |
| myfile.o | This indicates the presence of an object file. |
| myfile.pdf | This indicates a file in Portable Document Format (PDF). |
| myfile.ps | This denotes a PostScript file. |
| myfile.tex | This denotes TEX formatting programme input. |
| myfile.txt | This denotes a standard text file. |
| myfile.zip | This indicates that the archive has been compressed. |

# File Structure

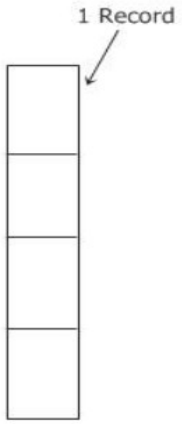
**There are 3 types of the File Structure**

## File Structure 1




The file in this instance is an unstructured sequence of bytes, as can be seen in the figure that is located above. Because of this, the operating system is completely oblivious to the contents of the file because all it can see are the bytes that make up the file.

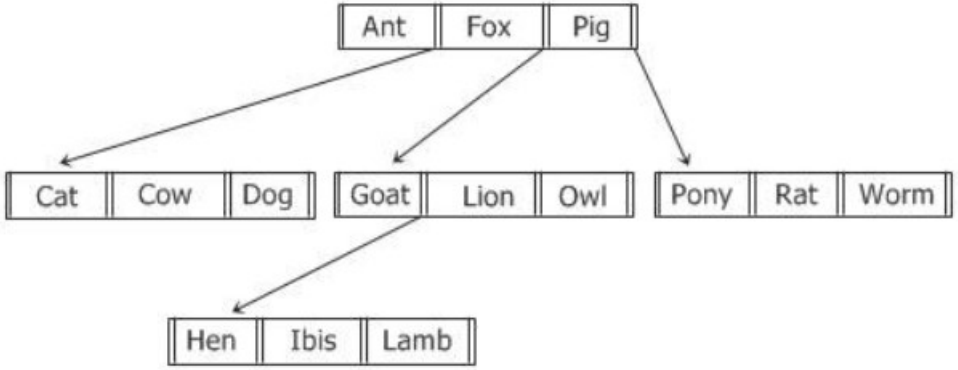
## File Structure 2




File is defined as a sequence of records of a fixed length, each of which has some kind of internal structure.

The concept that a read operation brings back a record and that a write operation merely adds a record to the end of the file is fundamental to the concept that a file is a sequence of records

## File Structure 3




The second structure of a file, where a file is defined as a sequence of records of a fixed length, each of which has some kind of internal structure.

The concept that a read operation brings back a record and that a write operation merely adds a record to the end of the file is fundamental to the concept that a file is a sequence of records

There are several types of files supported by many operating systems. For example, a Windows-based operating system supports the following types of files.

And UNIX-based operating systems support the following types of files:

- Directories
- Regular files
- Character special files
- Block special files

**Regular Files**

Regular files contain user information. Generally, regular files are either ASCII files or binary files. These two types of files are described below.

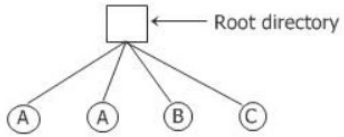
**ASCII Files**: ASCII files basically consist of lines of text. The big plus point or advantage of ASCII files is that they can be displayed and printed as is, and they can also be edited using any text editor such as Notepad, Wordpad, Notepad+, etc.

**Binary Files**: The binary files listed on the printer produce random garbage. Generally, binary files have some internal structure that is only known to the programs that use them. Technically, binary files are just a sequence of bytes, and the OS will only execute a file if it has the proper format

Directories are the system files that are used to maintain the structure of the file system.

**Single-Level Directory Systems**

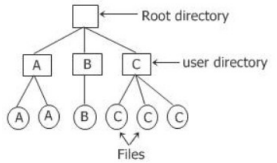
The simplest form of the directory system is a directory that contains all of the files, also known as the root directory. This system was common in earlier PCs because there was only one user.




Having only one directory in a system with multiple users may result in different users accidentally using the same names for their files.

**Two-Level Directory Systems**

Two-level directory systems are used to avoid the problems caused by the single-level directory system, as we have learned in the previous section.




In two-level directory systems, give each user a private directory.

So, in this two-level directory system, the names chosen by one user don't affect the names chosen by another user, and having the same name in two or more directories doesn't cause any problems

## Character Special Files

Character special files are mostly about I/O and are used to model serial input/output devices like printers, networks, and so on.

## Block Special Files

Basically, block-specific files are used to model the disks

| file type | usual extension | function |
|---|---|---|
| executable | exe, com, bin or none | ready-to-run machine-language program |
| object | obj, o | compiled, machine language, not linked |
| source code | c, cc, java, pas, asm, a | source code in various languages |
| batch | bat, sh | commands to the command interpreter |
| text | txt, doc | textual data, documents |
| word processor | wp, tex, rtf, doc | various word-processor formats |
| library | lib, a, so, dll | libraries of routines for programmers |
| print or view | ps, pdf, jpg | ASCII or binary file in a format for printing or viewing |
| archive | arc, zip, tar | related files grouped into one file, sometimes compressed, for archiving or storage |
| multimedia | mpeg, mov, rm, mp3, avi | binary file containing audio or A/V information |

There are three ways to access a file into a computer system:
Sequential-Access,
Direct Access,
Index sequential Method.

***Sequential-Access***

It is the simplest access method. Information in the file is processed in order, one record after the other. This mode of access is by far the most common

- Data is accessed one record right after another record in an order.
- When we use read command, it move ahead pointer by one
- When we use write command, it will allocate memory and move the pointer to the end of the file

**Direct Access –**
Another method is *direct access method* also known as *relative access method*.

A fixed-length logical record that allows the program to read and write record rapidly. in no particular order.

The direct access is based on the disk model of a file since disk allows random access to any file block. For direct access, the file is viewed as a numbered sequence of block or record. Thus, we may read block 14 then block 59, and then we can write block 17.

**Sequential Access**

```
read next
write next
reset
```

no read after last write
(rewrite)

**Direct Access –** file is fixed length logical records

```
read n
write n
position to n
    read next
    write next
rewrite n
```

n = relative block number

**Index sequential method –**

It is the other method of accessing a file that is built on the top of the sequential access method.

These methods construct an index for the file.

The index, like an index in the back of a book, contains the pointer to the various blocks

# File Attributes

A file has a name and data. Moreover, it also stores meta information like file creation date and time, current size, last modified date, etc. All this information is called the attributes of a file system.

Here, are some important File attributes used in OS:

**Name:** It is the only information stored in a human-readable form.

**Identifier**: Every file is identified by a unique tag number within a file system known as an identifier.

**Location:** Points to file location on device.

**Type:** This attribute is required for systems that support various types of files.

**Size**. Attribute used to display the current file size.

**Protection**. This attribute assigns and controls the access rights of reading, writing, and executing the file.

**Time, date and security:** It is used for protection, security, and also used for monitoring

# File Operations

**File Open Operation**
The file open operation is used to open the file. The file can be opened to read, modify, or for any other purpose. You can double-click on a file to open it.

**File Close Operation**
The file must be closed to free up the internal table space when all the accesses are finished and the attributes and disc addresses are no longer needed.

**File Read Operation**
The file read operation is only used to read the data stored in the specified file.

**File Write Operation**
The file write operation is used to save data to a file, usually at the current position.

**File Append Operation**
The file append operation is similar to the file write operation, except that it only adds data to the end of the file.

**File Seek Operation**
A method is required for random access files to specify where to take the data. As a result, the file seek operation handles this task.

**File Get Attribute Operation**
The file get attributes operation is performed by processes when they need to read the file attributes in order to complete their tasks.

**File Set Attribute Operation**
The file set attribute operation is used to set some of the attributes (user settable attributes) after the file has been created.

**File Rename Operation**
The file rename operation is used to change the name of an existing file.

The Implementation of the files can be done in two structures

On-disk

In-memory structures

**Boot Control Block**

Boot Control Block contains all the information which is needed to boot an operating system from that volume. It is called boot block in UNIX file system.

**Volume Control Block**

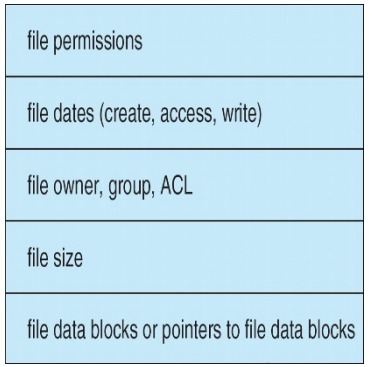
Volume control block all the information regarding that volume such as number of blocks, size of each block, partition table, pointers to free blocks and free FCB blocks. In UNIX file system, it is known as super block. In NTFS, this information is stored inside master file table.

**Directory Structure (per file system)**

A directory structure (per file system) contains file names and pointers to corresponding FCBs. In UNIX, it includes inode numbers associated to file names.

**File Control Block**

File Control block contains all the details about the file such as ownership details, permission details, file size,etc. In UFS, this detail is stored in inode. In NTFS, this information is stored inside master file table as a relational database structure. A typical file control block is shown in the image below.



**In-memory Mount Table**

In-memory mount table contains the list of all the devices which are being mounted to the system. Whenever the connection is maintained to a device, its entry will be done in the mount table.

**In-memory Directory structure cache**

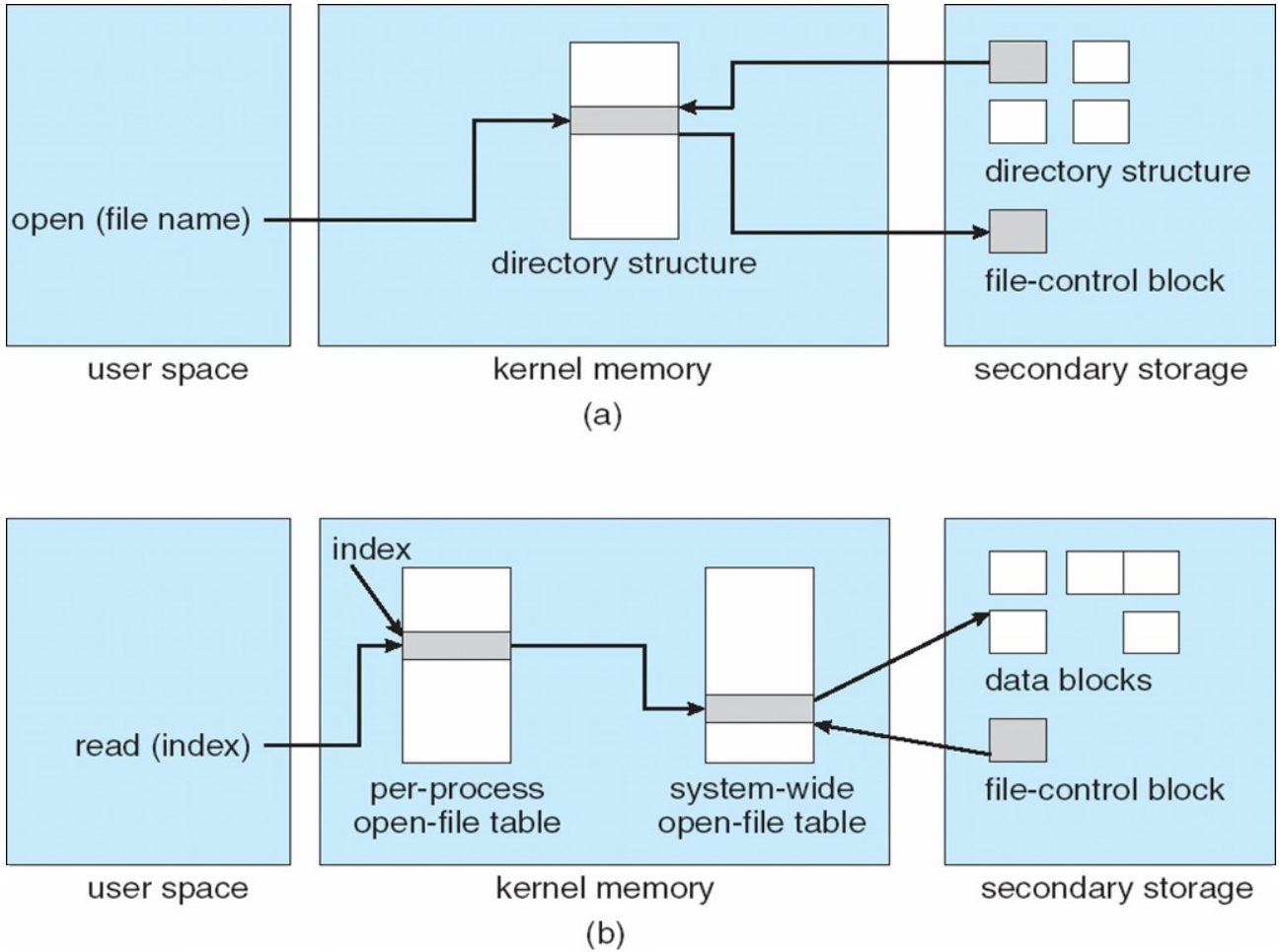
This is the list of directory which is recently accessed by the CPU. The directories present in the list can also be accessed in the near future so it will be better to store them temporally in cache.

**System-wide open file table**

This is the list of all the open files in the system at a particular time. Whenever the user open any file for reading or writing, the entry will be made in this open file table.

**Per process Open file table**

It is the list of open files subjected to every process. Since there is already a list which is there for every open file in the system therefore It only contains Pointers to the appropriate entry in the system wide table


open (file name)
directory structure
directory structure
file-control block
user space
kernel memory
secondary storage
(a)
index
read (index)
per-process open-file table
system-wide open-file table
data blocks
file-control block
user space
kernel memory
secondary storage
(b)

# End of Chapter Unit-5