Files

Introduction...

- Secondary storage is the non-volatile (*persistent* repository for (both user and system) data and programs.
- As (integral or separate) part of an operating system, the file system manages this information on secondary storage.
- Uses of secondary storage include storing various forms of programs (source, object, executable) and temporary storage of virtual memory pages (paging device or swap space).
- Information in secondary storage may be in a variety of forms, including readable text and raw data (e.g., binary).

File concept...

- A file is a *named* collection of related information, usually as a sequence of bytes, with two views:
    - *Logical (programmer's) view*, as the users see it.
    - *Physical (operating system) view*, as it actually resides on secondary storage.
- What is the difference between a file and a data structure in memory? Basically,
    - files are intended to be non-volatile; hence in principle, they are long lasting (persistent),
    - files are intended to be moved around (i.e., copied from one place to another), accessed by different programs and users, and so on.

Where do file (systems) live?...

Anywhere long-term data lives:

- Hard drives/Solid state drives
- Flash (USB) drives
- CDs/DVDs/BlueRays
- BUT, data is stored in different ways.

File attributes...

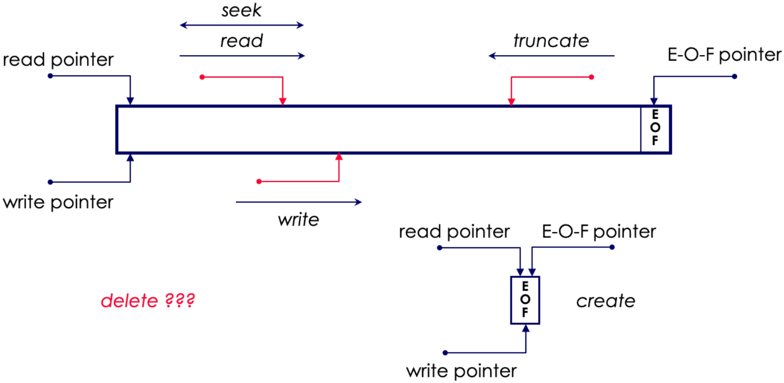
Each file is associated with a collection of information, known as *attributes*:

- NAME, owner, creator
- type (e.g., source, data, binary)
- location (e.g., I-node or disk address)
- organization (e.g., sequential, indexed, random)
- access permissions
- time and date (creation, modification, and last accessed)
- size
- variety of other (e.g., maintenance) information.

File operations...

There are six basic operations for file manipulation:

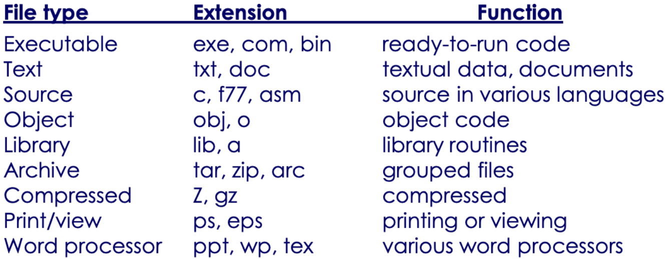
create, write, read, delete, reposition r/w pointer (a.k.a. *seek*, and *truncate* (not very common).



File types...

A common implementation technique (as organizational help with consistent usage) is to include the type as an extension to the file name:

create, write, read, delete, reposition r/w pointer (a.k.a. *seek*, and *truncate* (not very common).



File type	**Extension**	**Function**
Executable	exe, com, bin	ready-to-run code
Text	txt, doc	textual data, documents
Source	c, f77, asm	source in various languages
Object	obj, o	object code
Library	lib, a	library routines
Archive	tar, zip, arc	grouped files
Compressed	Z, gz	compressed
Print/view	ps, eps	printing or viewing
Word processor	ppt, wp, tex	various word processors

Files are structured internally to meet the expectations of the program(s) that manipulate them.

File access methods...

The information stored in a file can be accessed in a variety of methods:

- *Sequential*: in order, one record after another.
- *Direct (random)*: in any order, skipping the previous records.
- *Keyed*: in any order, but with particular value(s); e.g., hash table or dictionary. *TLB lookup* is one example of a keyed search.

Other access methods, such as *indexed*, can be built on top of the above basic techniques. IBM's indexed sequential access method (ISAM) is built on random and sequential access.

Directories...

A directory is a *symbol table*, which can be searched for information about the files. Also, it is the fundamental way of organizing files. Usually, a directory is itself a file.

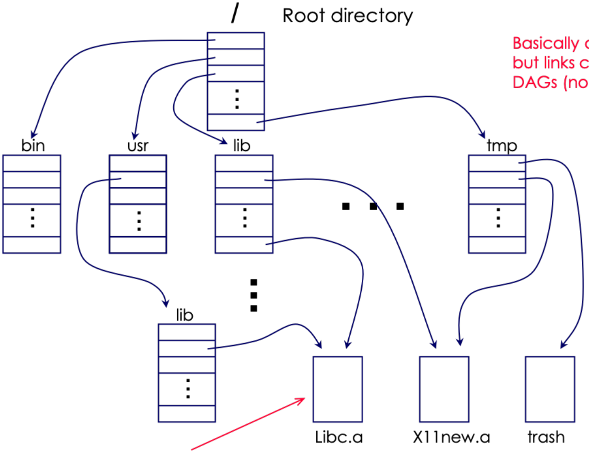
A typical *directory entry* contains information (attributes) about a file. Directory entries are added as files are created, and are removed when files are deleted.

Common directory structures are:

- **Single-level (flat)**: shared by *all* users.
- **Two-level**: one level for *each* user.
- **Tree**: arbitrary (sub)-tree for *each* user

An example: UNIX directories...

UNIX uses an advanced form of tree structure, known as directed acyclic-graph (DAG) directory.



Basically a tree,
but links convert to
DAGs (no cycles!)

/usr/lib/libc.a **or** /lib/libc.a

File sharing...

Allowing users to share files raises a major issue: *protection*.

A general approach is to provide *controlled access* to files through a set of operations such as read, write, delete, list, and append. Then permit users to perform one or more operations.

One popular protection mechanism is a condensed version of access list, where the system recognizes three classifications of users with each file and directory:

- user
- group
- other