File Systems

CS204 – Operating Systems

By:

Dilum Bandara

Outline

- Why file systems?
- Files
  - Naming, types, structure, access, attributes, operations
- Directories
  - Single level, two level & hierarchical directory systems
  - Operations
- Path Names
- File systems
  - DOS file system, FAT32, NTFS, ext

Why file systems?

- The data must survive after the termination of the process using it
- It must be possible to store very large amount of data
- Multiple processes must be able to access data concurrently

- Solution is to store those data in units called files on disks & other media

Files

- The data stored in a file must be persistent
- Files are managed by the OS
- How they are
  - structured, used, protected & implemented are major concerns
- The part of the OS that deals with files is known as the File System

Files - Naming

- The exact rules depends based on the OS
- However most of them allow files to be:
  - 1-8 characters
  - Digits & several selected symbols
  - Modern ones supports up to 255 characters
  - Examples
    - osslides, osslides1, osslides2, osslides-1, urgent!
- Some file systems are case sensitive
  - DOS, Windows – Case insensitive
  - UNIX, Linux – Case sensitive

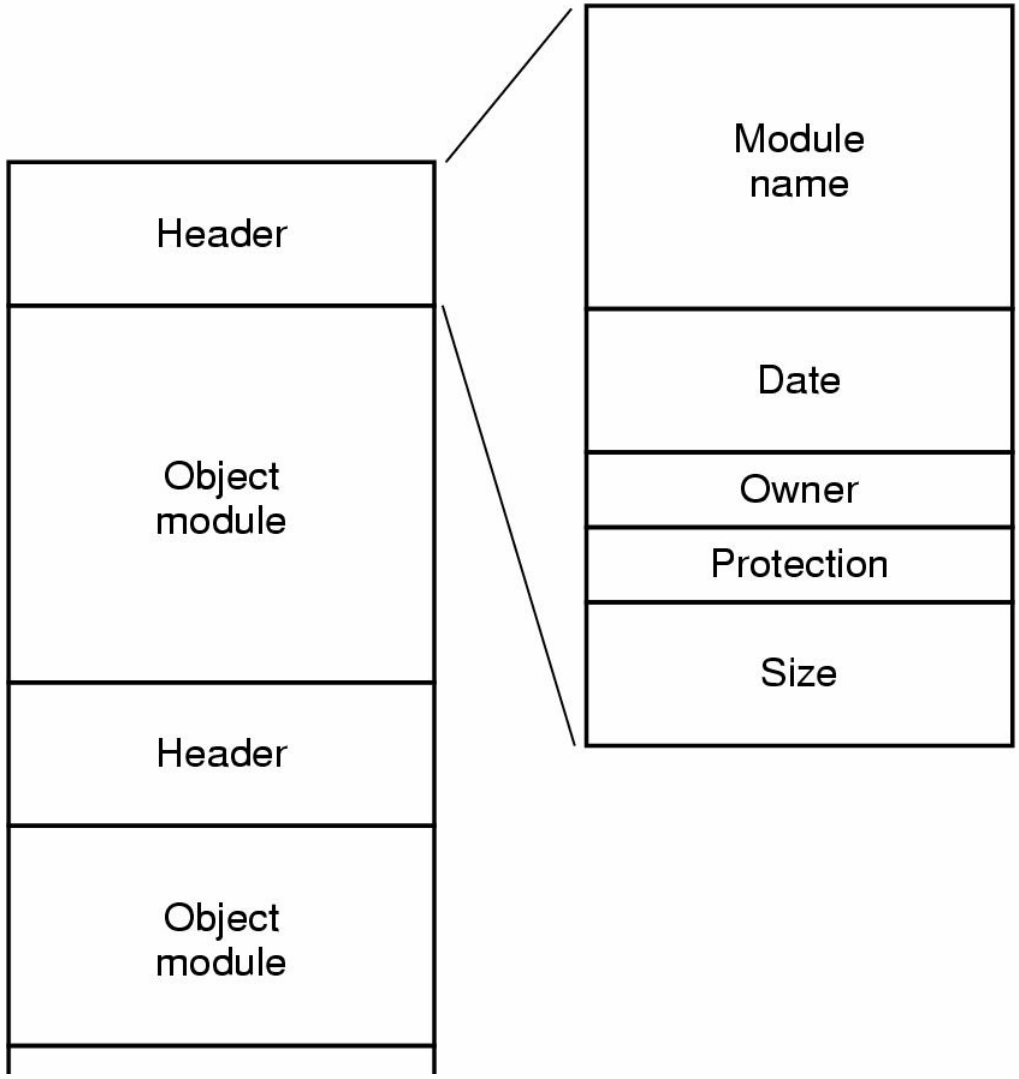
Files – Naming etc.

- Many OSs support two-part file systems
  - Parts are separated by a period (**.**)
  - Example: **<file name>.<extension>**
  - `test.txt, os.pdf, MyClass.java, prog.c`
- Extension indicates something about the file
- Not all OSs are aware of extensions
  - UNIX or Linux does not depend on the extension
  - But some applications may depend on the extension

Files – Types

- 2 major types
  - Regular files – ones that contain user data. These are either ASCII or binary
  - Directories – are systems files which are used to maintain the structure of the file system
- In UNIX also has
  - Character files – related to IO & used to model serial IO devices such as terminals & printers
    - `/dev/tty, /dev/lp, /dev/net`
  - Block files – are used to model disks
    - `/dev/hd1, /dev/hd2`

Files - Structure



File - Access

- Can be categorised as:

1. Sequential access
   - Read all the data starting from the beginning
   - Used in early days with magnetic tapes
   - Example: simple text files
2. Random access
   - Can read data in a file out of order
   - Were possible with the introduction of magnetic disks
   - Examples: Data bases, movies

File - Attributes

- A files includes set of other characteristics than just name & an extension
- some common attributes
  - Owner – current owner of file
  - Creator – ID of the person who created the file
  - Protection – who can access & who can't access
  - Read only flag – can it be modified or not
  - Hidden flag – display or not when listed
  - Archive flag – to be backed up or not
  - Last modified date, Created date, etc.

File - Operations

- Different systems allow operations to store & retrieve data from files
  - Create – create a new file with no data & set initial attributes
  - Delete – remove the file from system freeing up disk space
  - Open – before using a files must be open
  - Read – after opening a file data can be read
  - Write – after opening a file data can be written
  - Append - after opening a file data can be written to the end of the file
  - Close – when all the access is finished file must be closed

File – Operations cont…

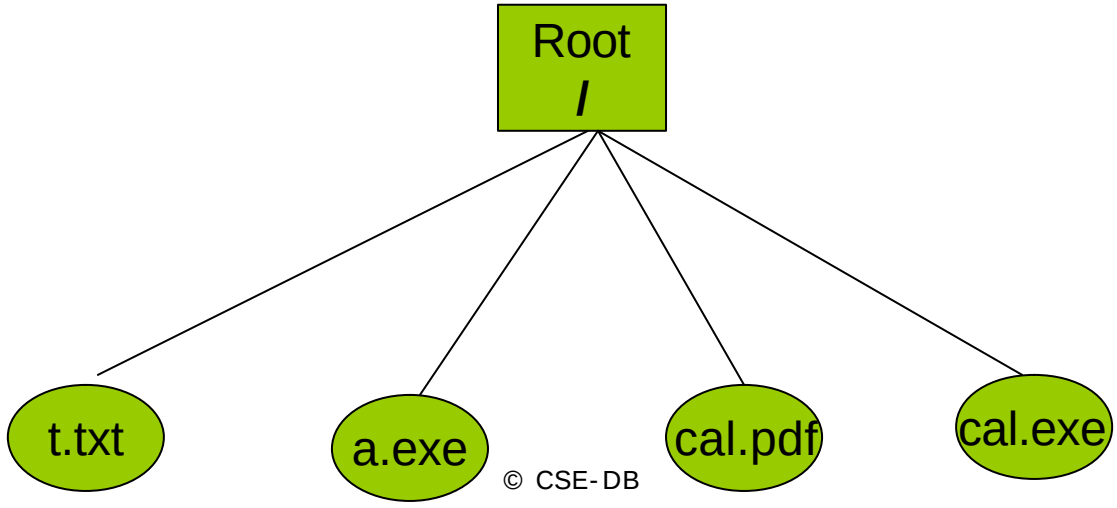
- Seek – used in random access a file
- Get attributes – get the attributes of a file
- Set attributes – set the attributes of a file
- Rename – change the name or the extension of a file

Directories

- Used to organise or keep track of files
- Are also called folder
- Most OSs consider even directories as files
  - DOS, UNIX, Linux call directories
  - While MS-Windows call them as Folders

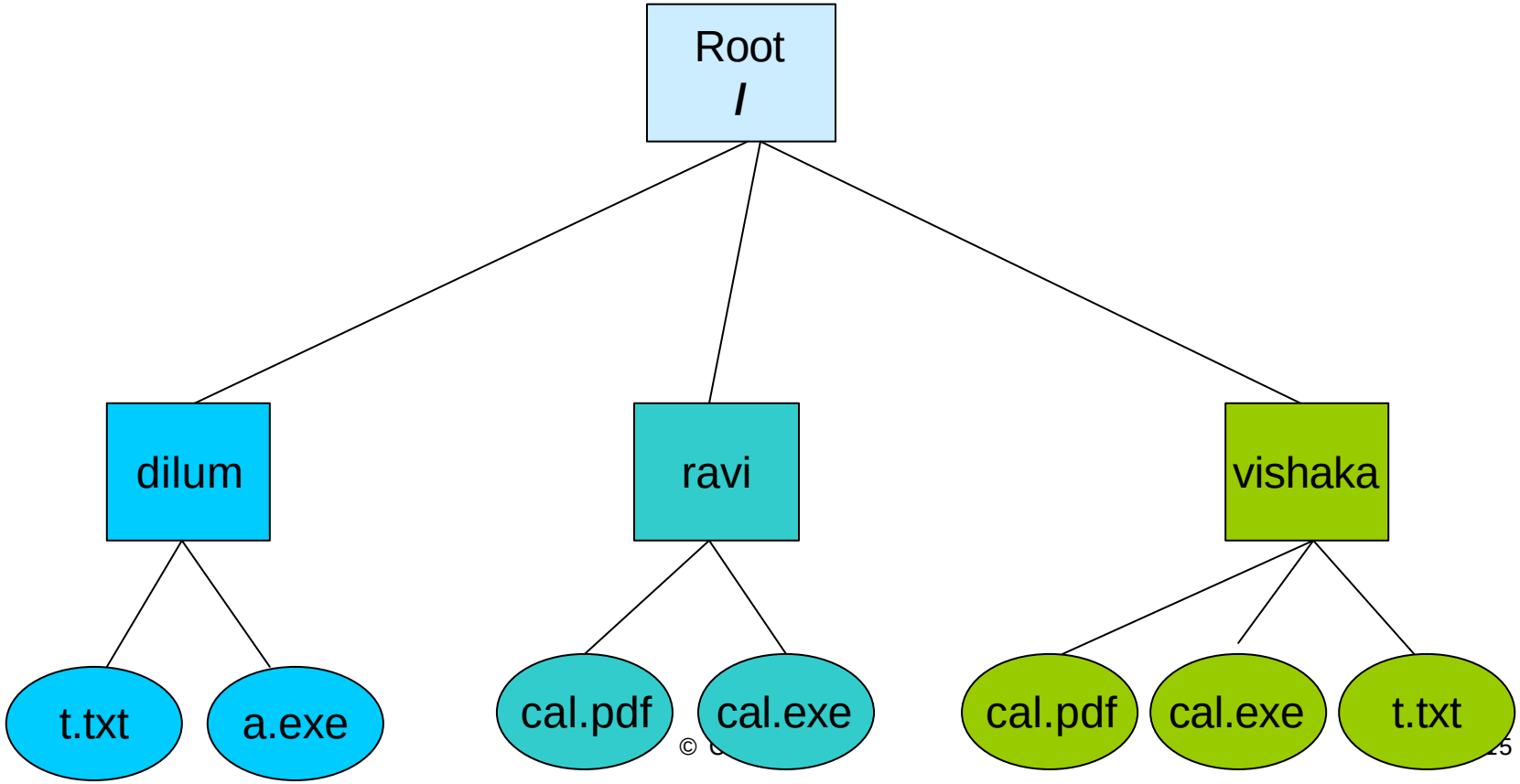
Directories - Single Level Systems

- Simplest form of directory system where 1 directory contains all the files
- This single directory is called the root
- Problems – in a multi-user systems uses can't have files with same name



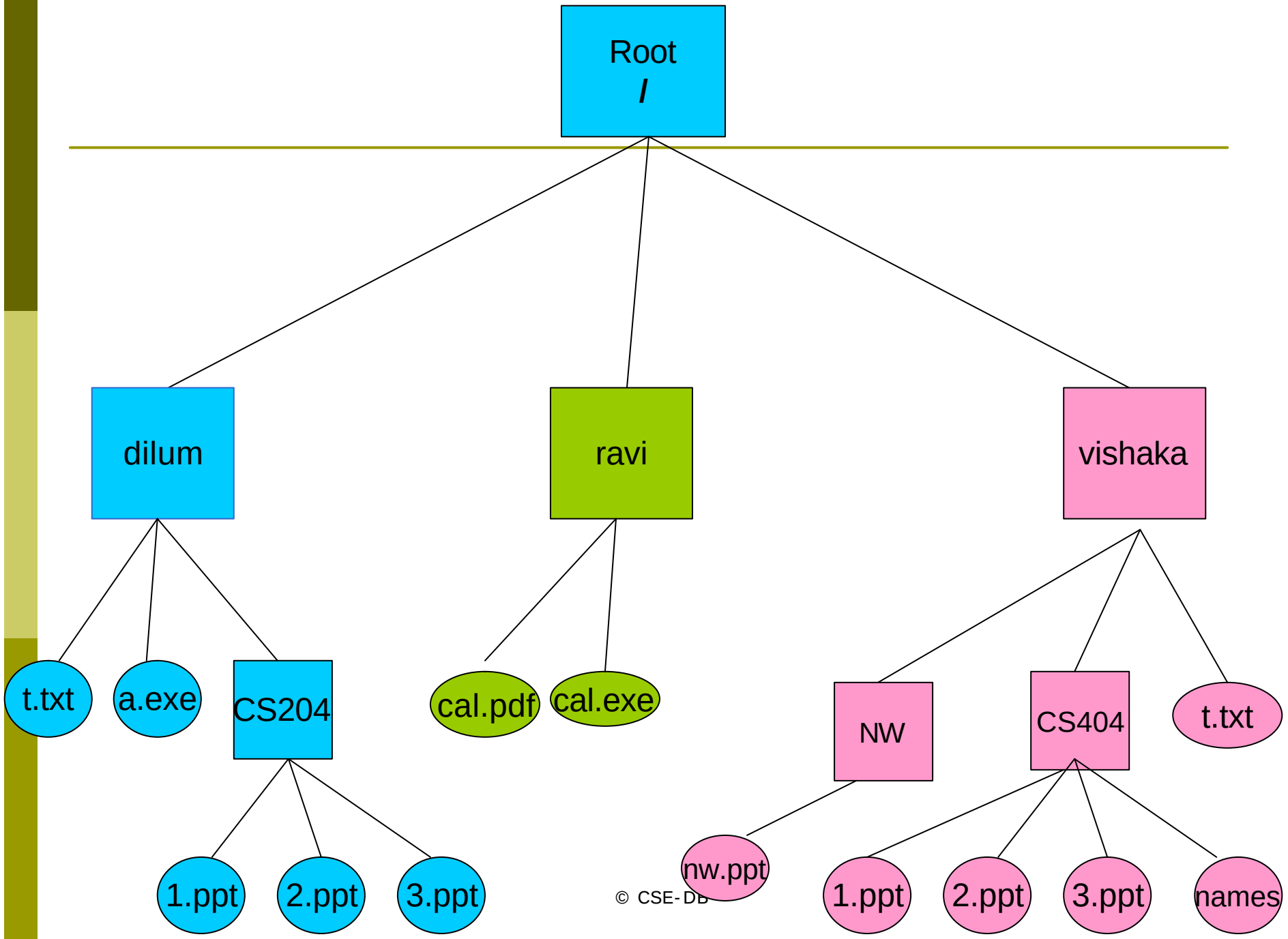
Directories - Two Level Systems

- To avoid the conflict each user is given a separate directory



Directories – Hierarchical Directory Structure

- Two Level directory structure is not enough when users want to manage their own files
- All most all the commercial OS supports multiple directory levels
- However CD-ROM file system has a limit in number of levels in the hierarchy


Root
/
dilum
ravi
vishaka
t.txt
a.exe
CS204
cal.pdf
cal.exe
NW
CS404
t.txt
1.ppt
2.ppt
3.ppt
nw.ppt
1.ppt
2.ppt
3.ppt
names

Directory Operations

- Create – create a new directory
- Delete – delete an existing directory
- Opendir – open the directory for reading
- Readdir – read the contents of the directory
- Closedir – close the directory
- Rename – change the name of the directory
- Link – allow files to appear in more then one directory. Related to file sharing

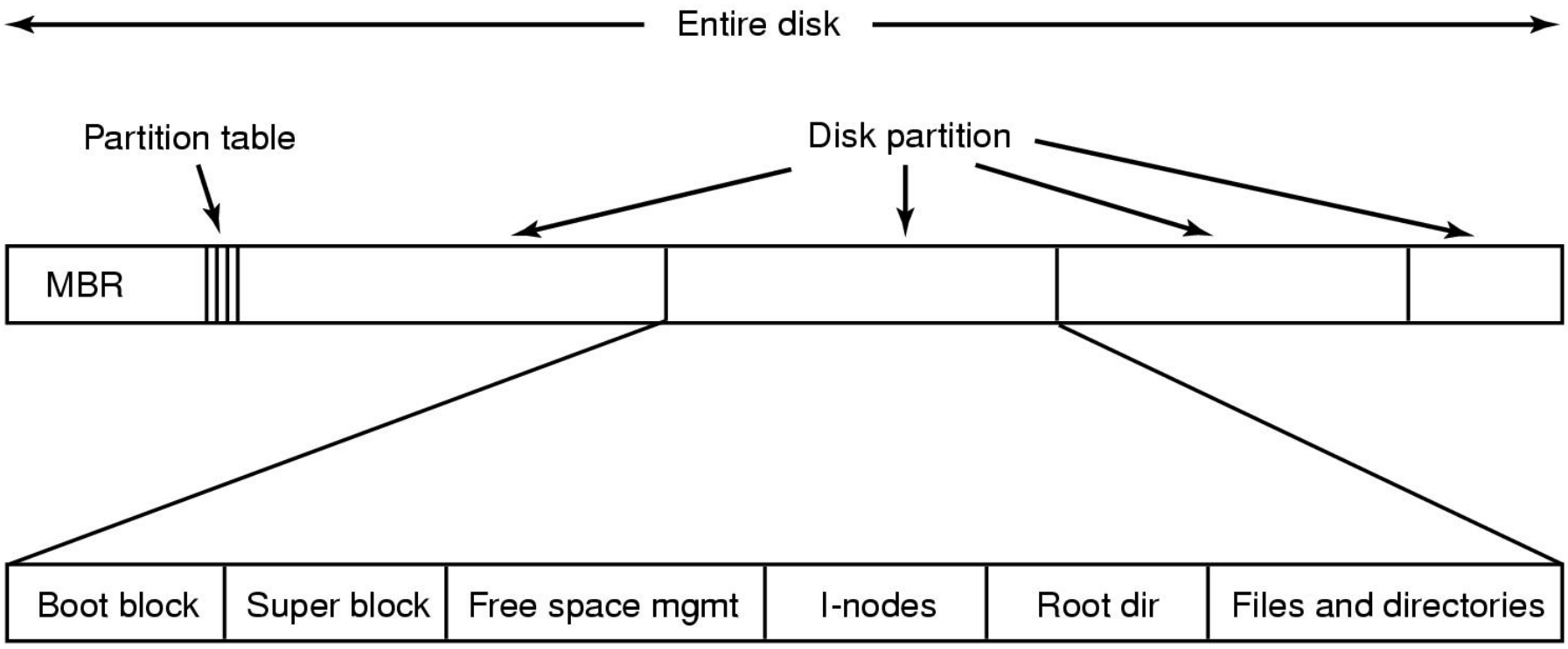
Path Names

- When files are in a directory tree there should be a mechanism to name them
- Absolute path names
  - Path from the root directory to the file
  - `/vishaka/NW/nw.ppt`
- Relative path names
  - Give relative to the current working directory
  - If currently in `NW` directory path name is `nw.ppt`
  - If currently in `vishaka` directory path name is `NW/nw.ppt`

Path Names cont…

- Regardless of the current working directory absolute pathnames will always work
- There are 2 special entries in each directory
  - `.` (dot) – refers to the current working directory
  - `..` (double dot/dotdot)– refers to the parent directory
  - Example : `./NW/nw.ppt`

File System layout



Courtesy of Modern Operating Systems, 2nd Edition, Andrew S. Tanenbaum

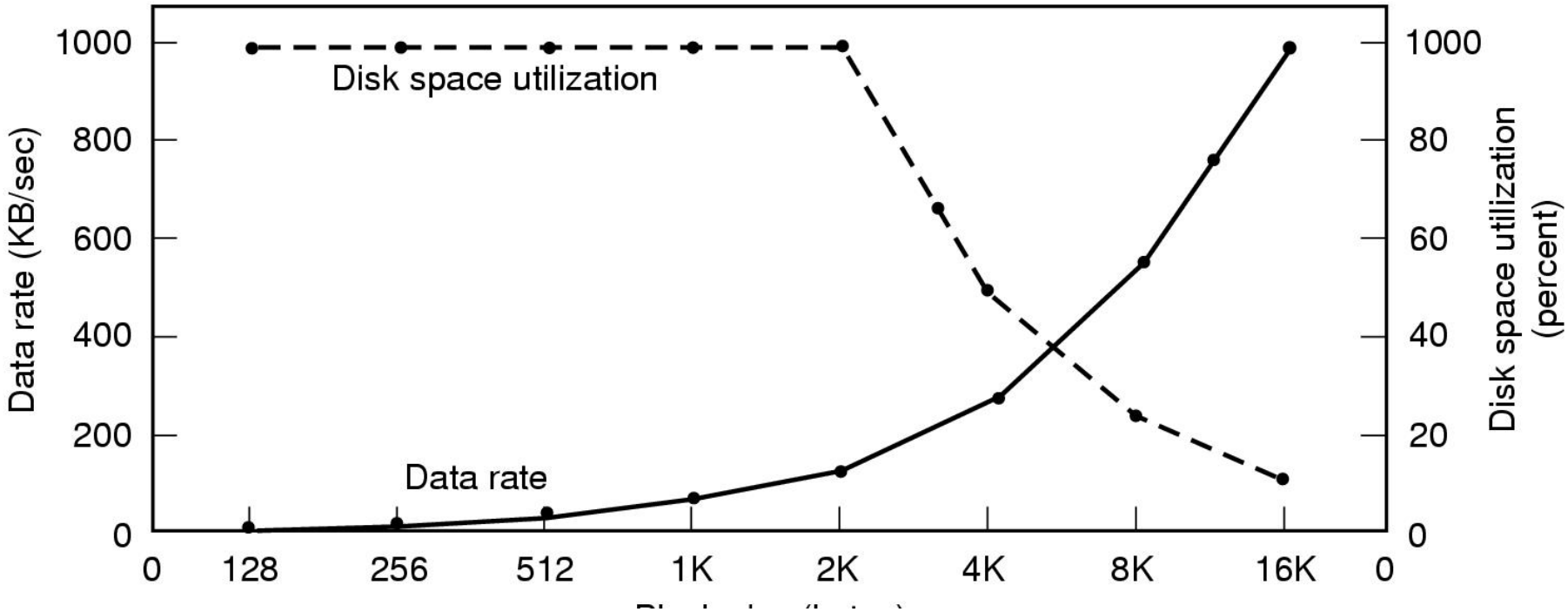
Implementing Files

- Need to keep track of where a file is located on the disk
- Files are stored as blocks
- Several approaches are used to store & keep track of files
  1. Contiguous allocation
  2. Link list allocation
  3. Link list allocation using a table in memory
  4. I –nodes

Block size

- For the easy of addressing & reading/writing data are access as fixed size blocks
- A single block can vary from the size of a single sector to multiple sector
- It should neither be too low nor large
- 4k is a good value

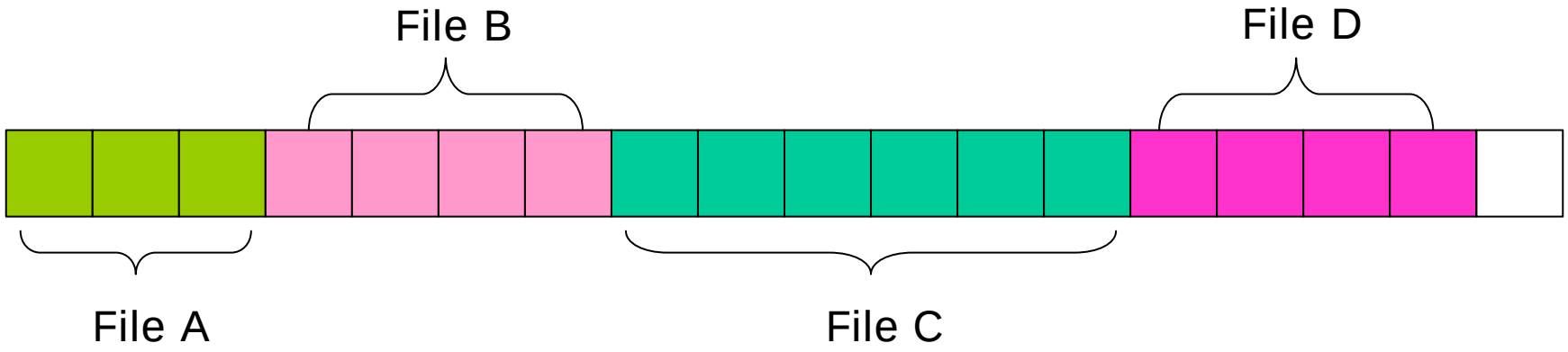
Effect of block size

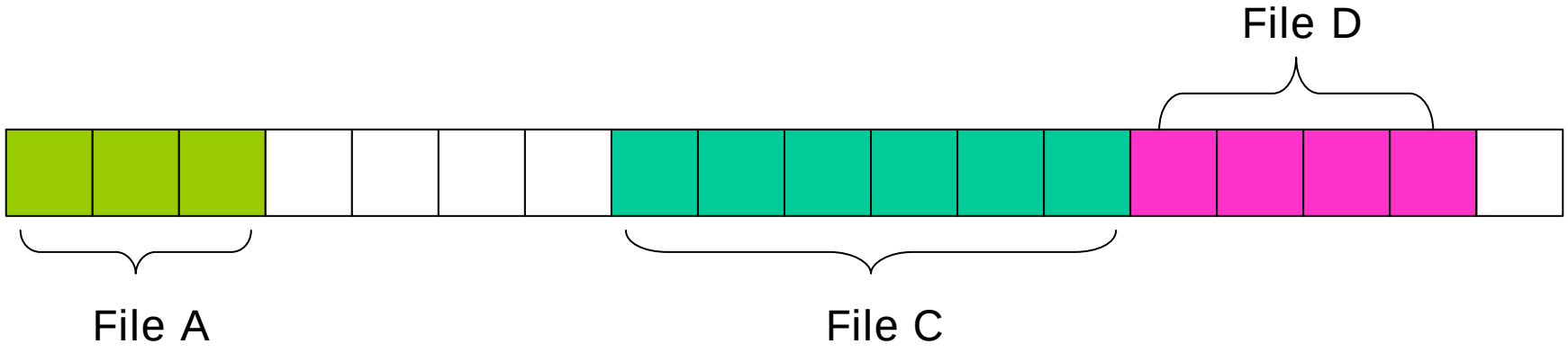


Higher block size → Higher data rate
Higher block size → Lower space utilization
Lower block size → Lower data rate
Lower block size → Higher space utilization

Courtesy of Modern Operating Systems, 2nd Edition, Andrew S. Tanenbaum

Contiguous Allocation cont…

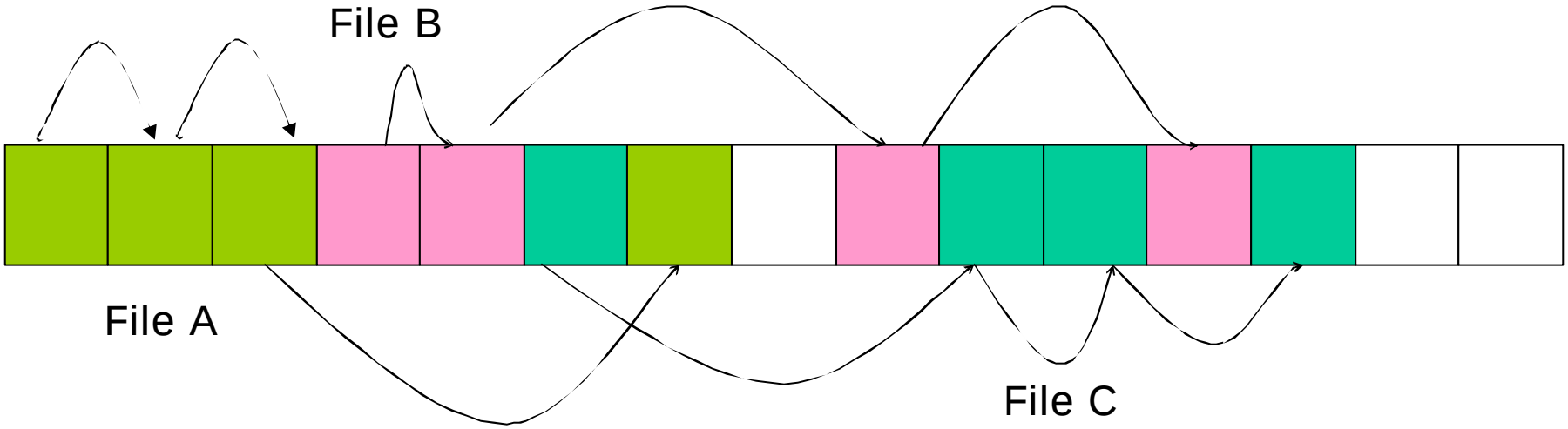


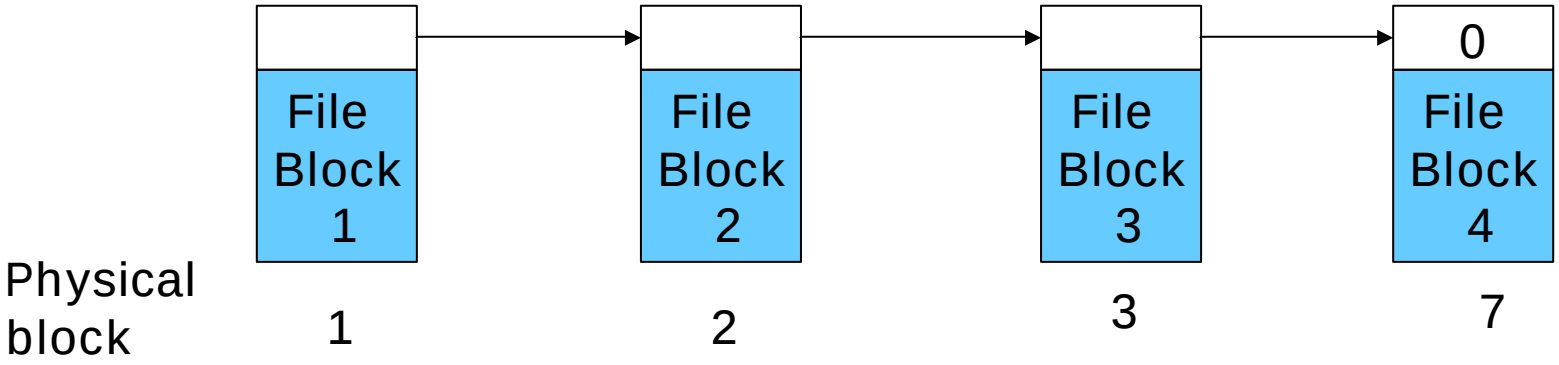


Contiguous Allocation

- Is the simplest allocation scheme
- One file is stored after
- Advantages
  - Simple to implement
  - Faster data reading
- Disadvantages
  - Disk fragmentation
- Not used in commercial OSs but suitable for Embedded OSs & CD-ROMs

Linked List Allocation

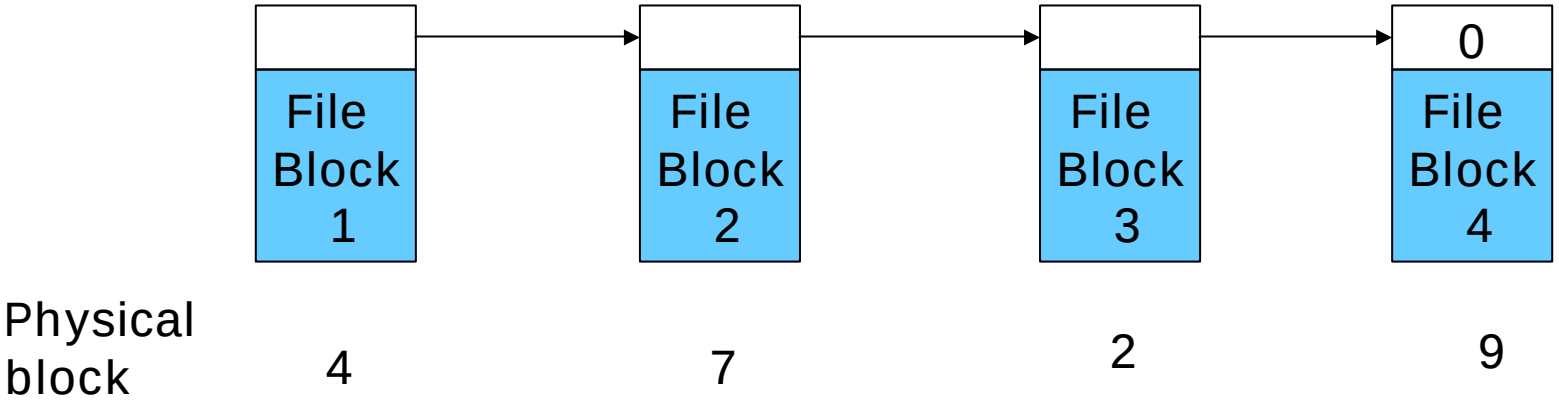


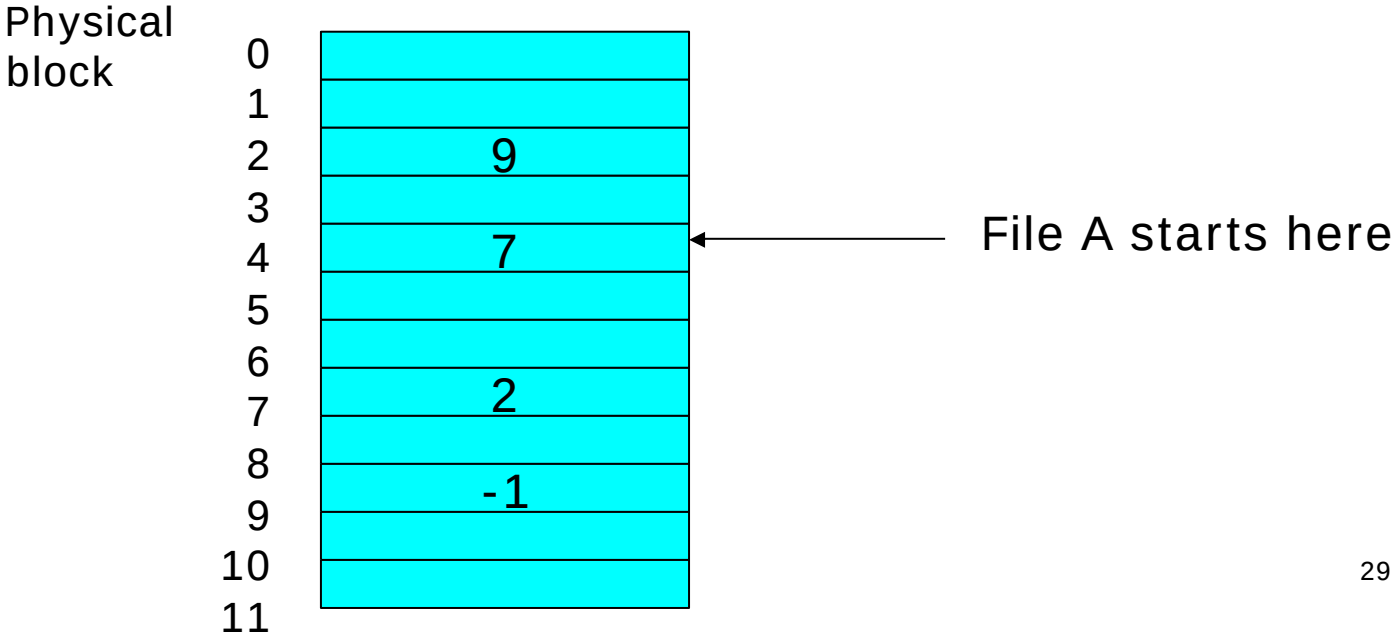


Linked List Allocation cont…

- Advantages
  - Every disk block can be used
  - No space is lost due to fragmentation
- Disadvantages
  - Random access is slow
  - Amount of data stored will not be powers of 2
  - If the link is lost rest of the file can’t be located

File Allocation Tables – FAT





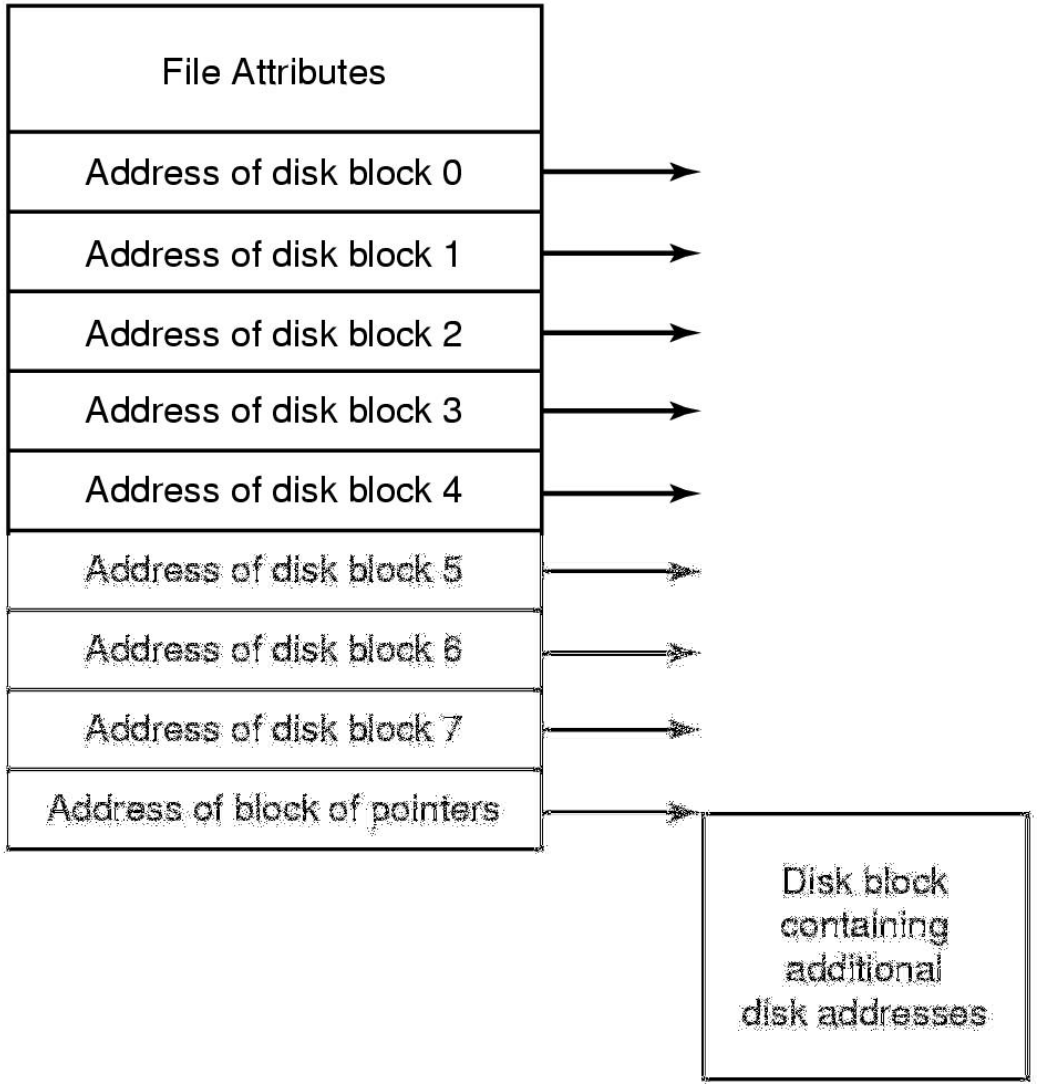
File Allocation Tables – FAT cont…

- Put the link list allocation entries in memory
- Advantages
  - Fast random access
  - If one of the blocks in the disk is lost still the rest of the file can be located
  - Full utilization of a single disk block
- Disadvantages
  - FAT takes some space in memory

I-nodes

- Index node is associated with each file
- Index node
  - i-node
- Given an i-node it's possible to find all blocks of the file

I-nodes cont…


File Attributes
Address of disk block 0
Address of disk block 1
Address of disk block 2
Address of disk block 3
Address of disk block 4
Address of disk block 5
Address of disk block 6
Address of disk block 7
Address of block of pointers
Disk block containing additional disk addresses

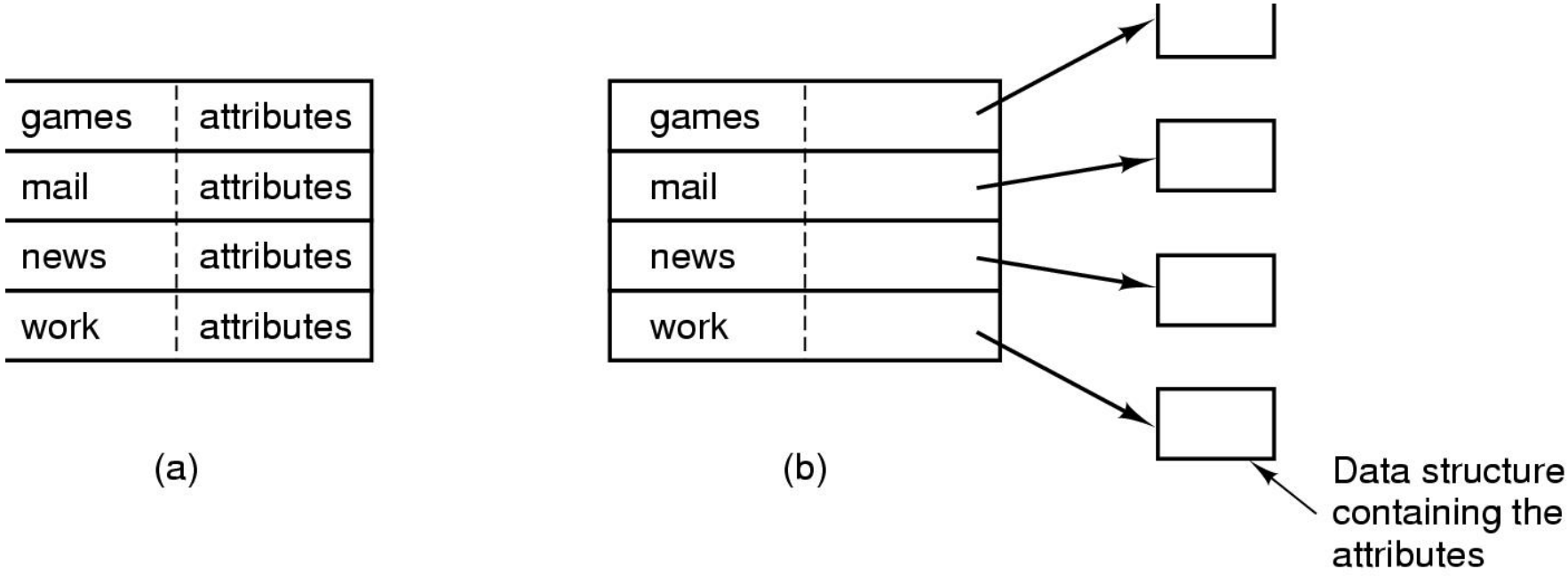
I-nodes cont…

- i-nodes are fixed in size
  - What about very large files?
- Advantages
  - When a file is open only the corresponding i-node should be in memory

Implementing directories

- The main function of a directory is to map the name of the file onto info needed to locate the data
- Finding appropriate disk blocks
  - Contiguous allocation
    - Disk address of entire file
  - Link list allocation
    - Number of the 1st block
  - i-node
    - Number of the i-node

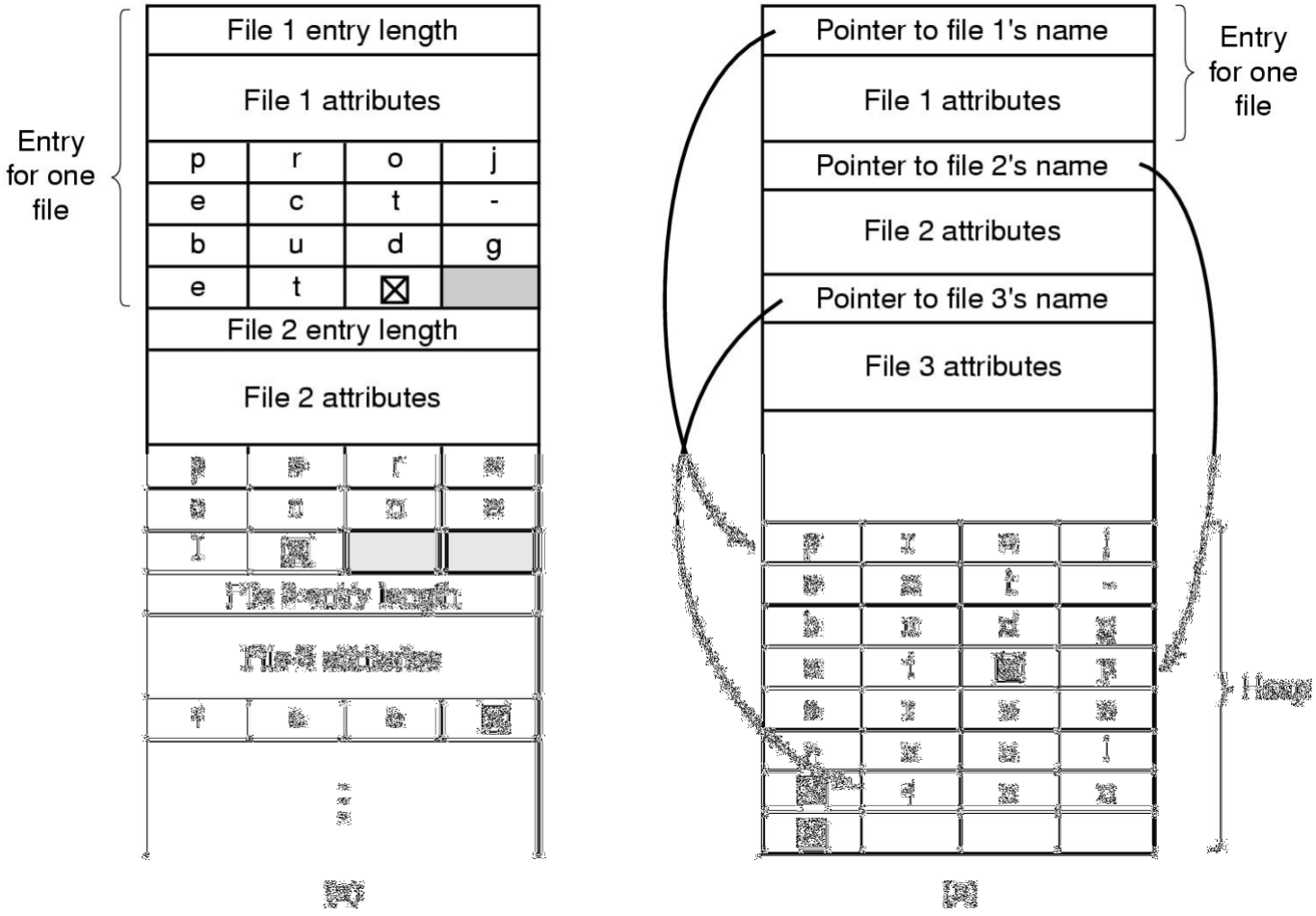
Storing file attributes



(a) Fixed size entries, disk addresses & attributes in directory entry

(b) Each entry just refers to an i-node

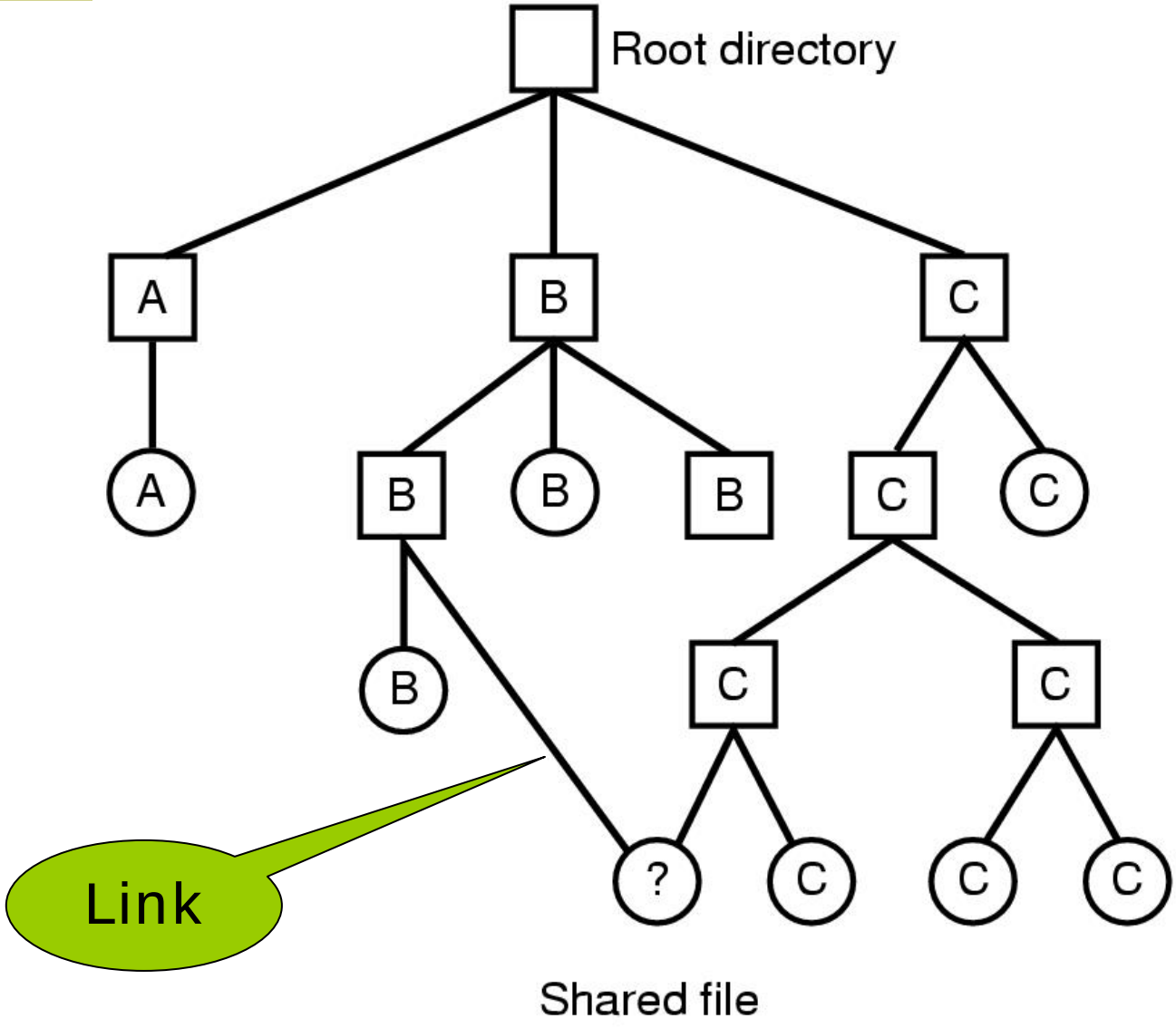
Handling long file names



(a) In-line
(b) In a heap

Courtesy of Modern Operating Systems, 2nd Edition, Andrew S. Tanenbaum

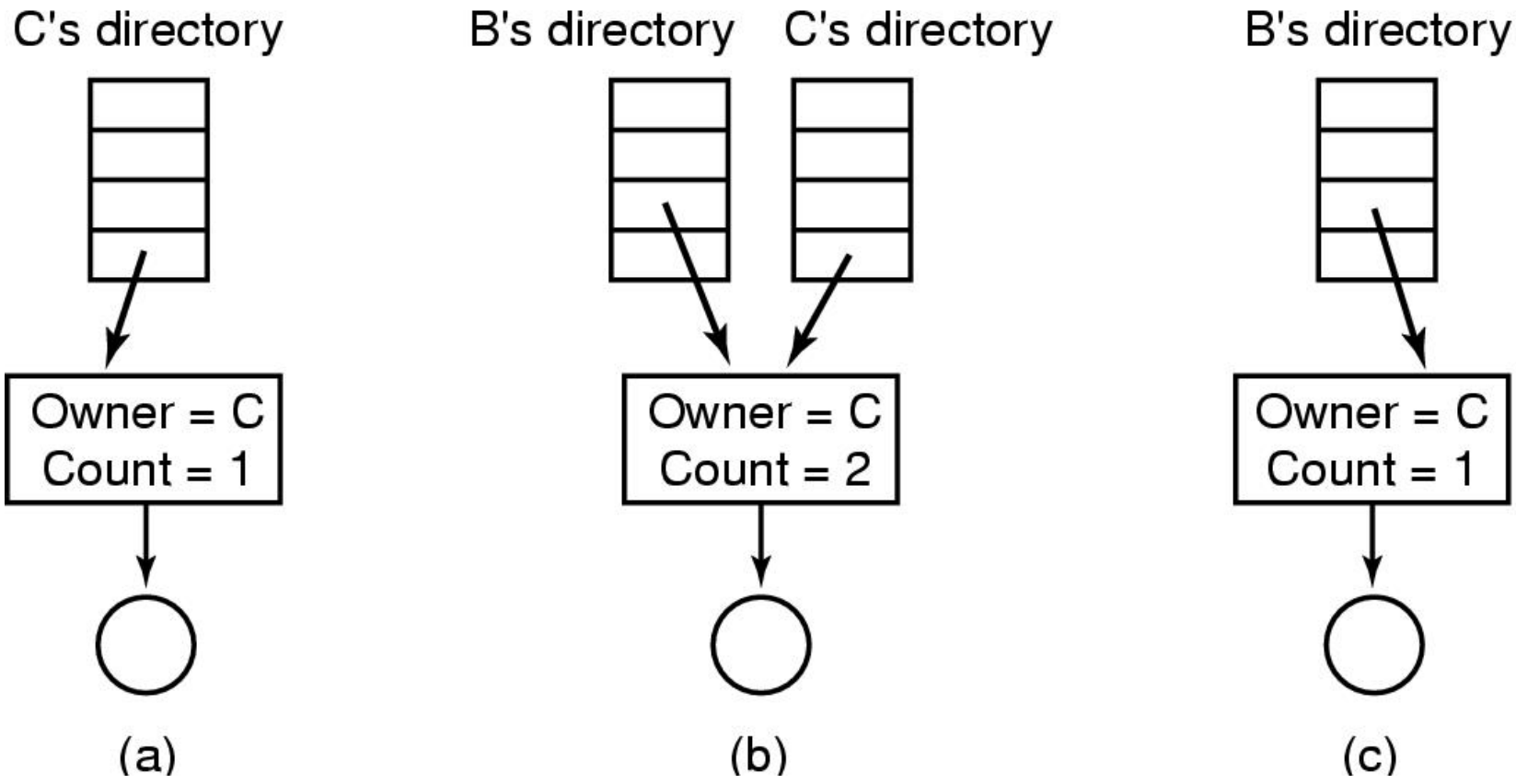
Shared files



Shared files cont…

- Need solutions to following problems
  - Whether to keep 2 copies of disk addresses?
  - If both users change the file how to synchronise?
- Solutions
  - Not to save disk block info on directories
    - Instead point to i-nodes
  - By creating new LINK file
    - Keep the path name to which it is linked
    - Referred as symbolic linking

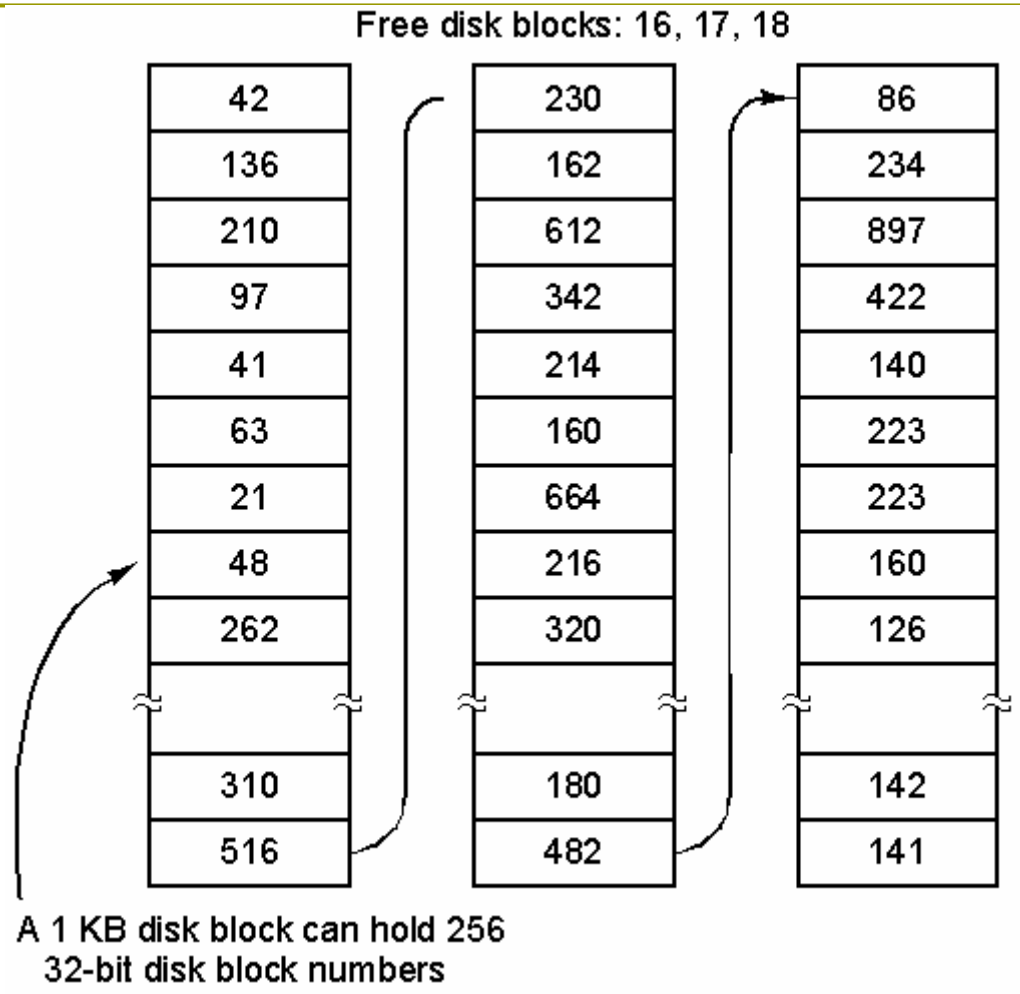
Shared files cont…



Shared files cont…

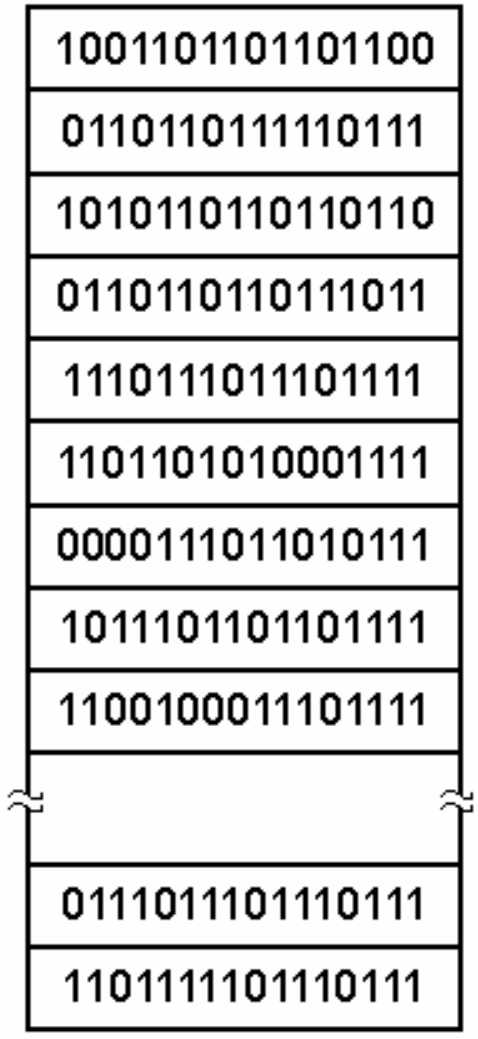
- If C removes the file B's i-node entry is wrong
  - So even when C removes the file it should remain
  - With C as the owner
  - Will be permanently deleted only if B release it
- This problem is not there with symbolic linking
  - If file deleted symbolic link is removed
  - Can be used to create links on machines all over the world

Tracking the free space



(a) Storing the free list on a linked list

Tracking the free space cont…

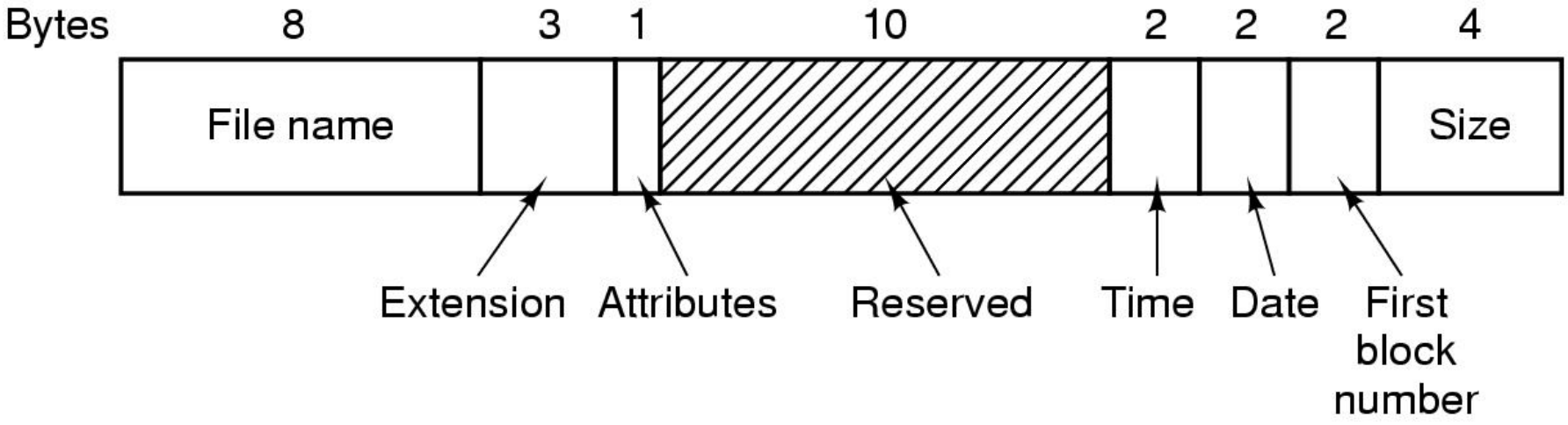


A bit map

DOS File System

- Make use of File Allocation Table (FAT)
- Use of 8+3 character file names
- Attributes
  - Read only, archived, hidden, system
- 2 versions
  - FAT-12
    - Max partition size 2MB
    - 4 separate partitions
  - FAT-16
    - Max partition size 2GB
    - 4 separate partitions

MS-DOS directory entry



Courtesy of Modern Operating Systems, 2nd Edition, Andrew S. Tanenbaum

MS-DOS

Maximum partition for different block sizes

Block size	FAT-12	FAT-16	FAT-32
0.5 KB	2 MB		
1 KB	4 MB		
2 KB	8 MB	128 MB	
4 KB	16 MB	256 MB	1 TB
8 KB		512 MB	2 TB
16 KB		1024 MB	2 TB
32 KB		2048 MB	2 TB

Courtesy of Modern Operating Systems, 2nd Edition, Andrew S. Tanenbaum

Windows 98 File System

- Actually came with Windows 95, 2nd release
- Namely FAT-32
  - Max partition size 2TB
  - More than 4 partitions
- File names up to 255 characters
- Was backward compatible with FAT-16

Windows 2000 File System

- NTFS – New Technology File System
- Initially used in Windows NT
- A single partition can be up to 2^{64} bytes
- File names can be up to 255 Unicode characters
- File system security is inbuilt
- Better performance & more reliable
- Not backward compatible with FAT-16 or FAT-32

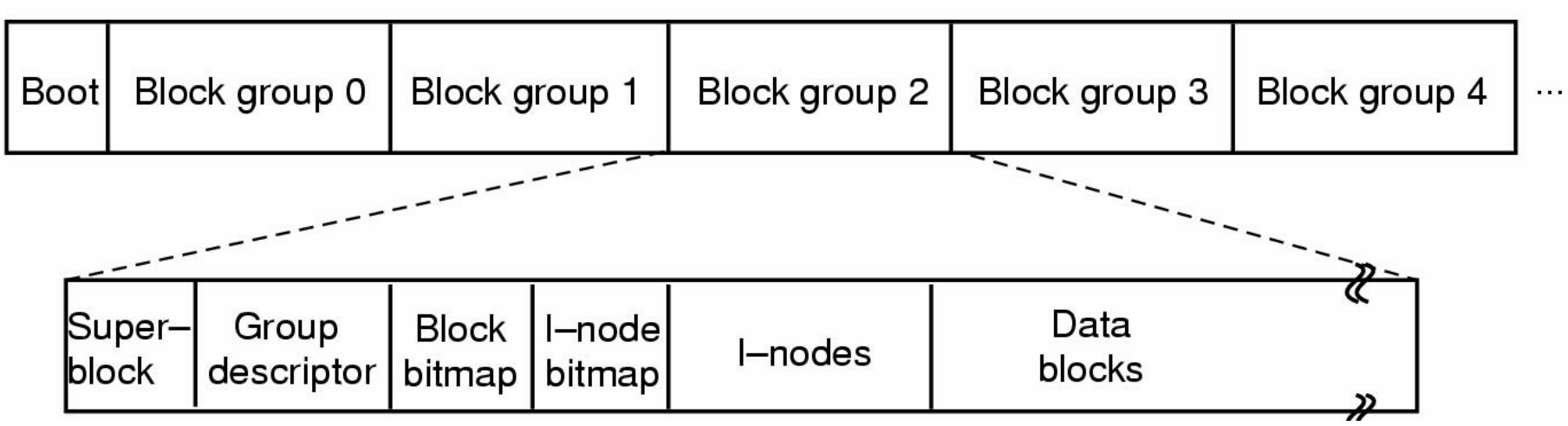
UNIX/Linux File Systems

- Make use of I-nodes
- Initially supported only 14 characters but later versions support 255 characters
- File system security is inbuilt
- Supports many file systems
  - V7
  - ext, ext2, ext3
  - NFS – Network File System
  - VFAT – UNIX version of FAT

Ext File System

- ext
  - Supports 14 characters
  - A single file can be up to 2GB
  - Later better versions were introduced such as ext2, ext3

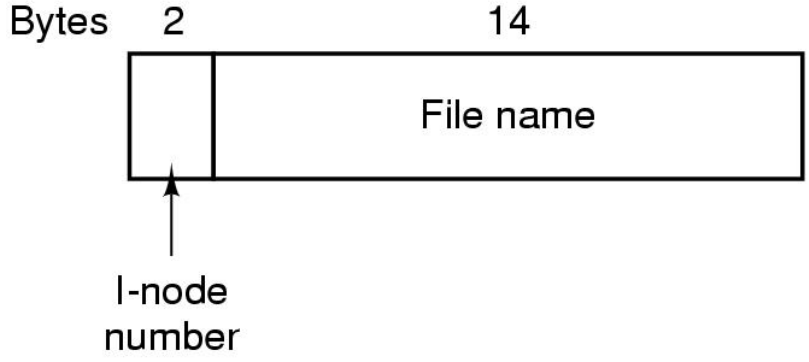
Linux file system



Courtesy of Modern Operating Systems, 2nd Edition, Andrew S. Tanenbaum

UNIX V7 file system

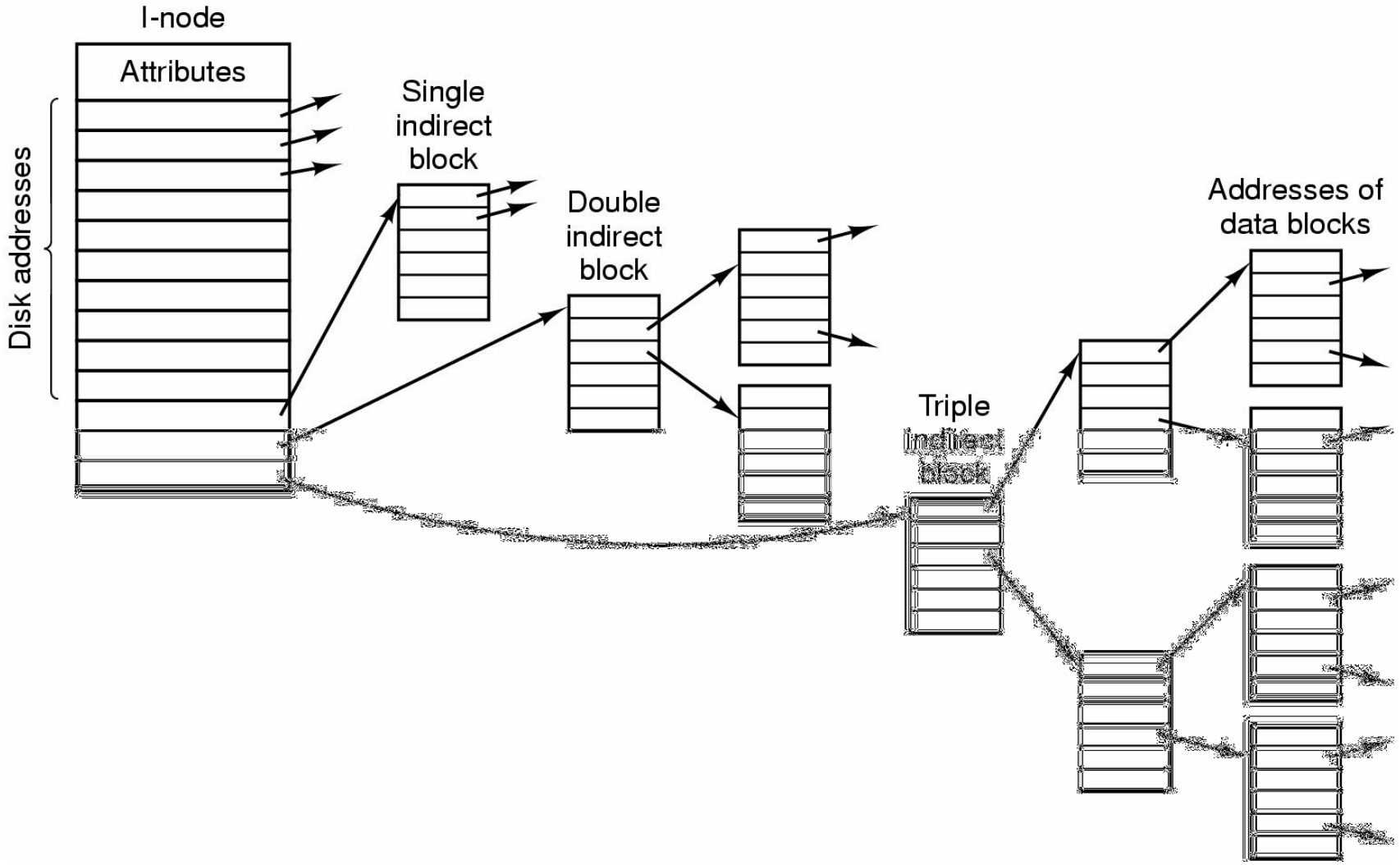
- A directory entry



- Number of i-nodes are limited 16^2

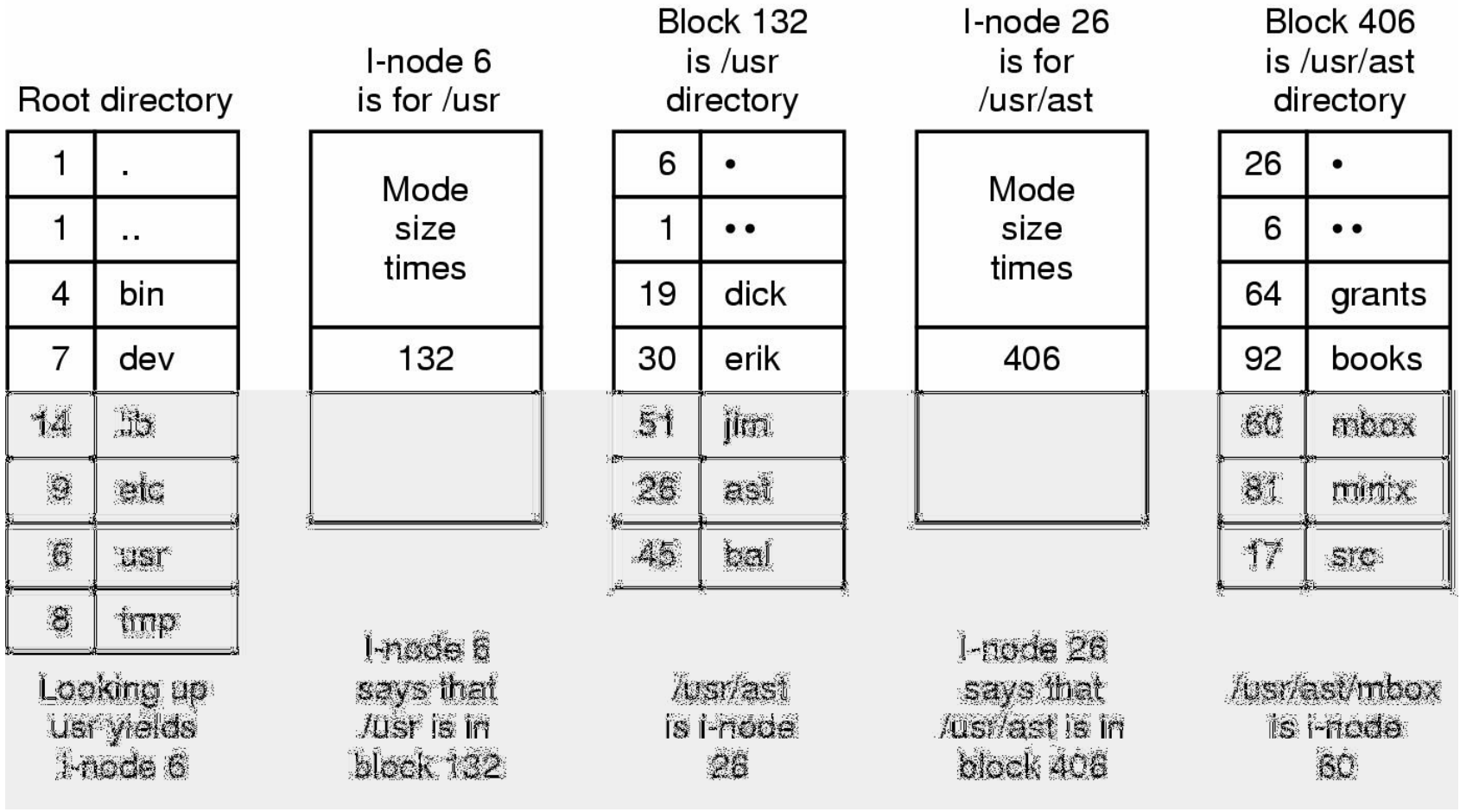
Courtesy of Modern Operating Systems, 2nd Edition, Andrew S. Tanenbaum

UNIX i-nodes



Courtesy of Modern Operating Systems, 2nd Edition, Andrew S. Tanenbaum

Locating a file `/usr/ast/mbox`



Courtesy of Modern Operating Systems, 2nd Edition, Andrew S. Tanenbaum

Summary

- What is a file & file system
- Files & Directories
- File naming, types, structure, access
- File & Directory attributes & operations
- Directory hierarchies
  - Single level, two level & hierarchical directory systems
- Path Names
- File systems
  - FAT-12, FAT-16, FAT-32, NTFS, ext