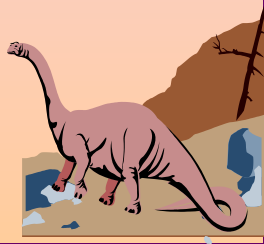
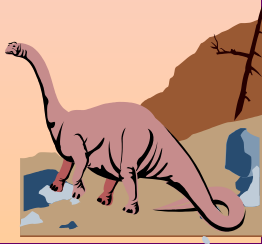
UNIT-IV: File-System

- Overview of Mass Magnetic Disks
- File Concept
- File Access Methods
- Directory Structure
- Disk space allocation methods
- Disk Scheduling

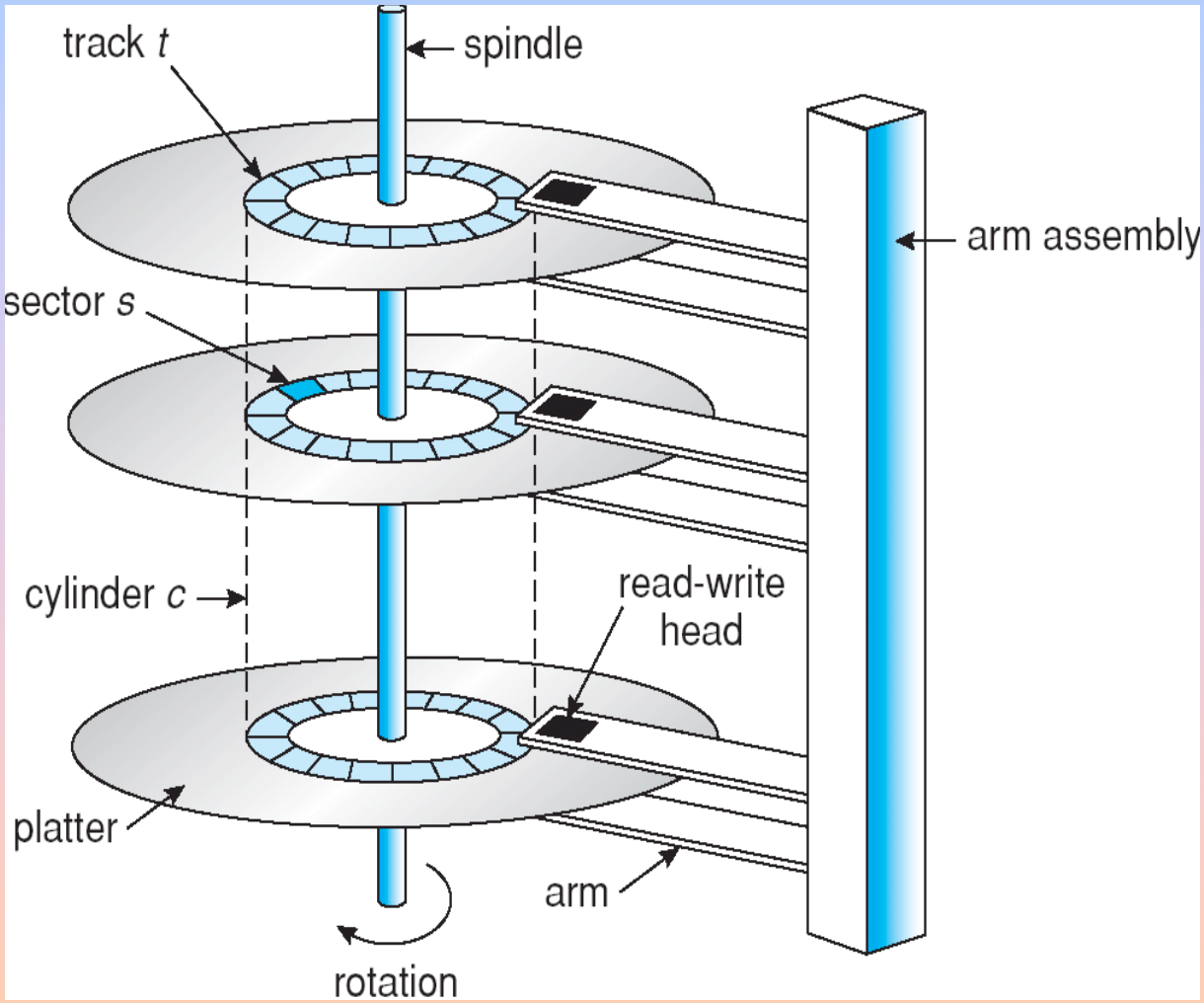


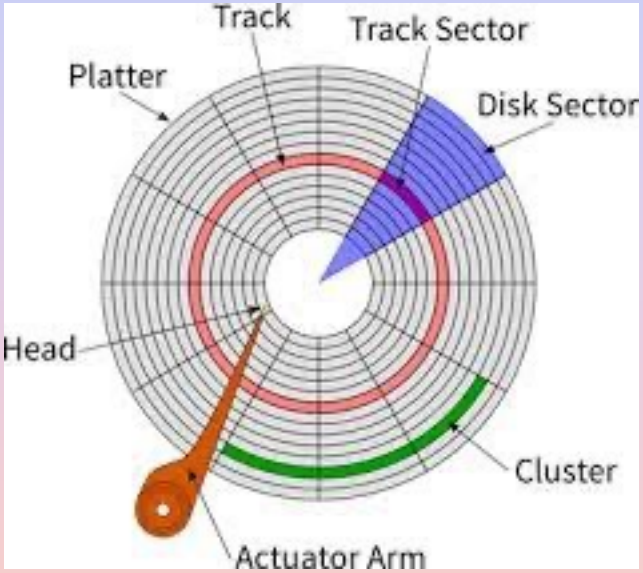
Overview of Mass Magnetic Disks

- Magnetic disks provide bulk of secondary storage of modern computers
  - Drives rotate at 60 to 200 times per second
  - Transfer rate is the rate at which data flow between drive and computer
  - Head crash results from disk head making contact with the disk surface
- Disks can be removable
- Drive attached to computer via I/O bus
  - Busses vary, including EIDE, ATA, SATA, USB, Fibre Channel, SCSI

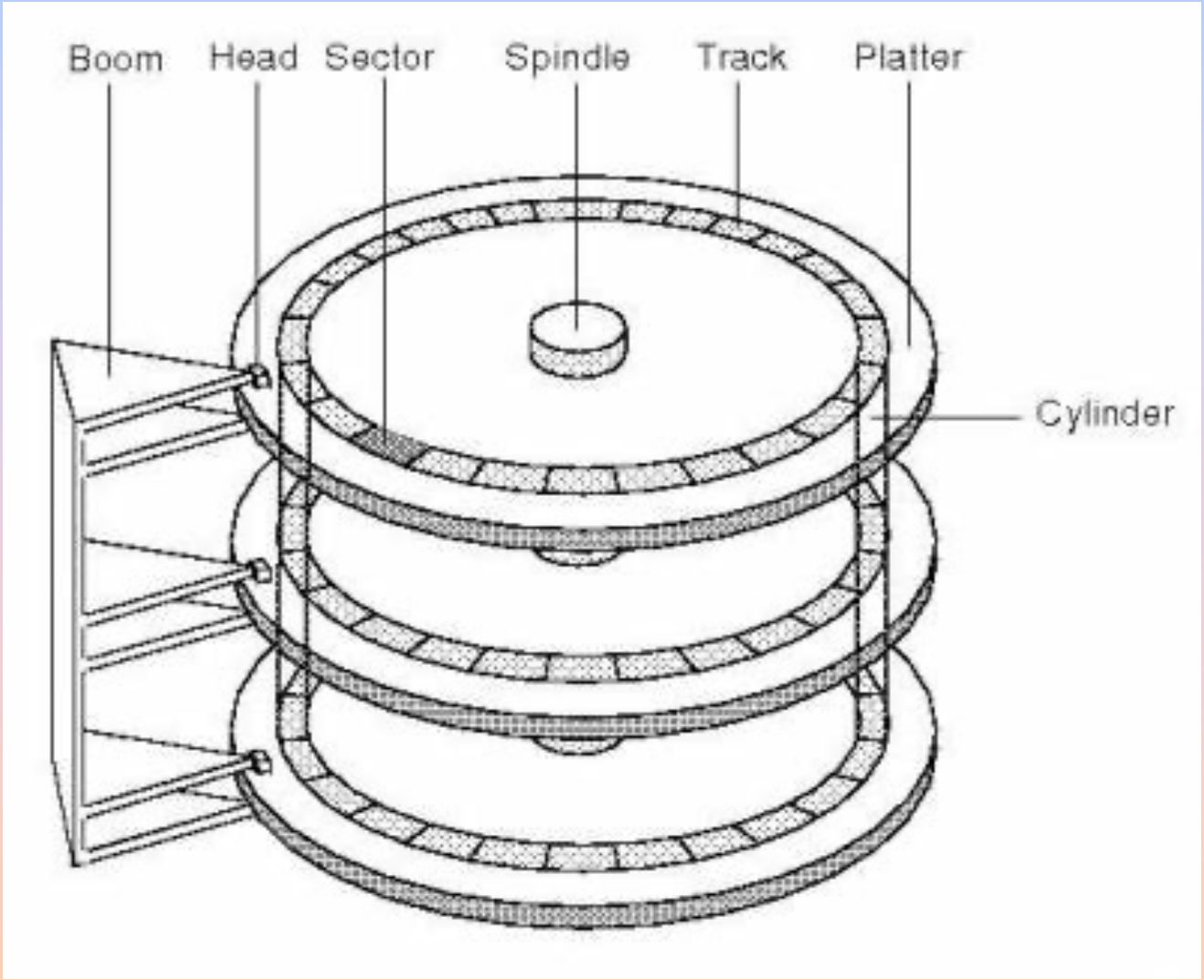


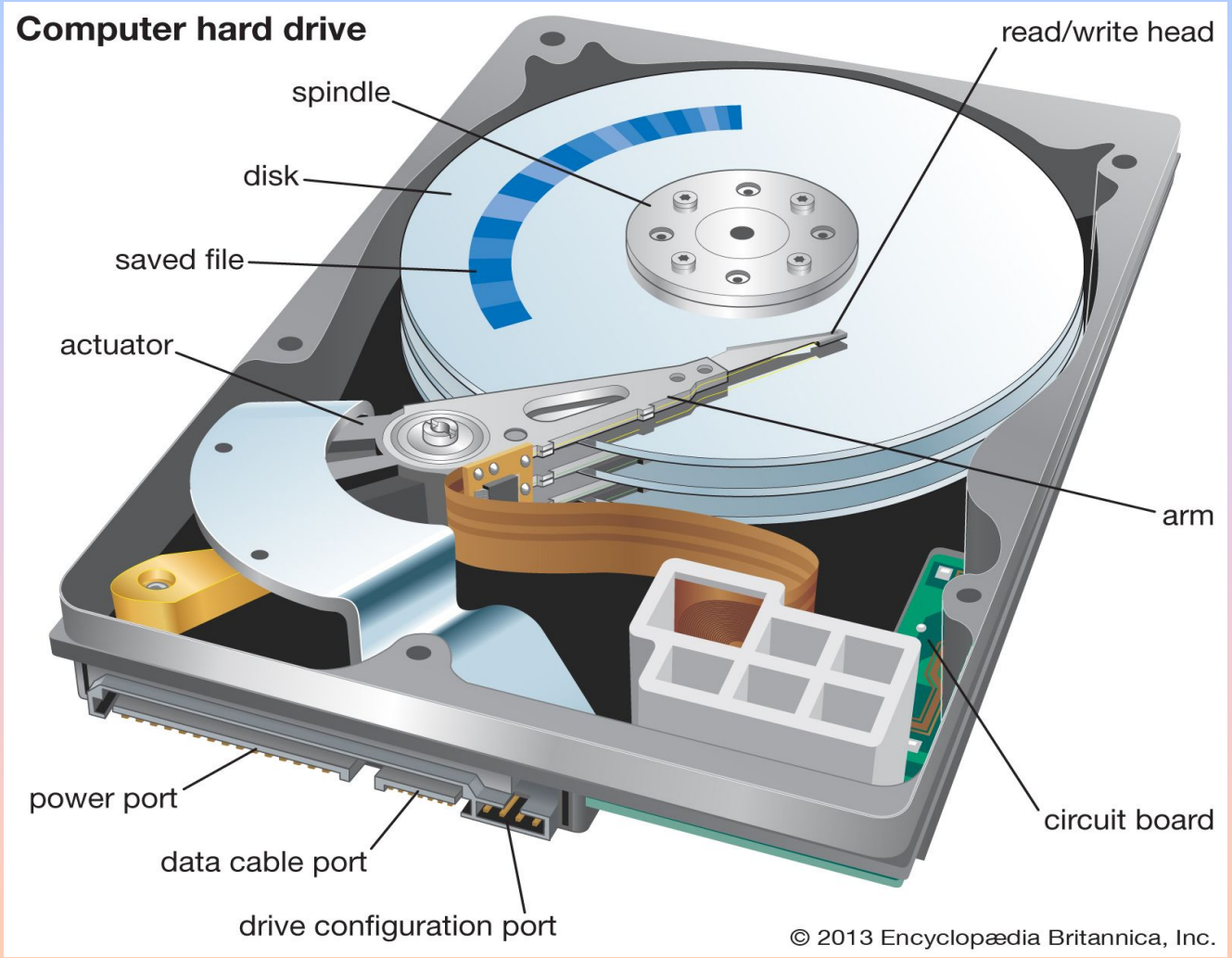
Moving-head Disk Mechanism





Important Disk Terminologies




Computer hard drive
read/write head
spindle
disk
saved file
actuator
arm
power port
circuit board
data cable port
drive configuration port
© 2013 Encyclopædia Britannica, Inc.

Track, Sector and Cluster/Block

Track:

- a track refers to a concentric circle on the surface of the disk where data is recorded.
- Typical width of a track is 200-250 nm

Sector:

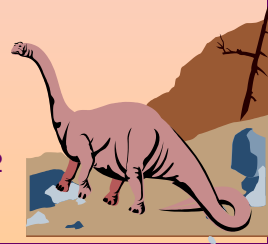
- A sector is the smallest unit of data storage on a hard disk.
- It is a fixed-sized portion of the disk's surface and is typically 512 bytes in size for most modern hard drives.

Cylinder:

- refers to a set of tracks that reside on multiple platters on the same radial position within a hard disk drive (HDD)

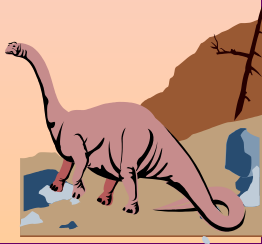
Cluster:

- A cluster, also known as a file allocation unit or block, is a group of contiguous sectors.
- It represents the smallest amount of disk space that can be allocated to store a file.



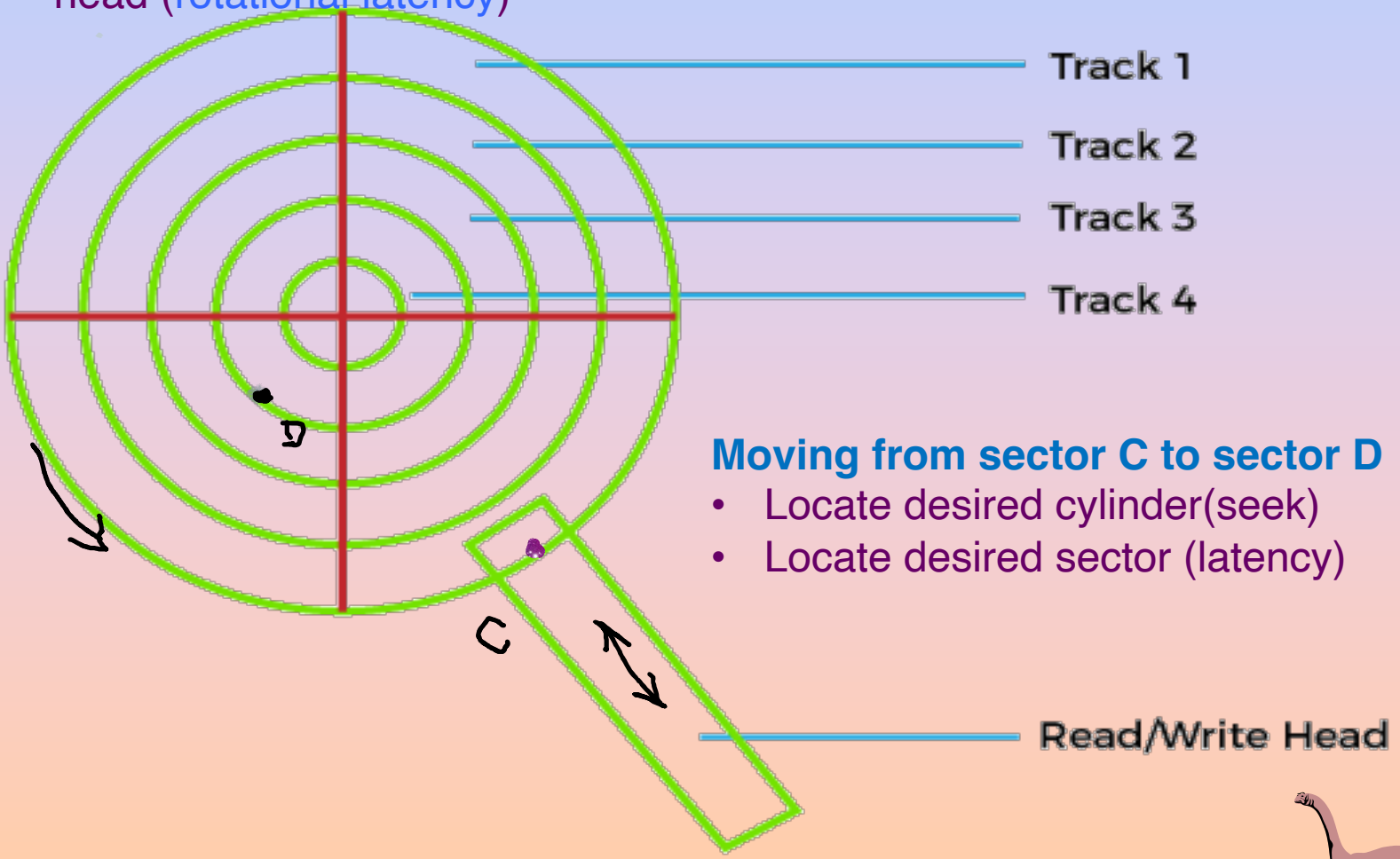
Disk Structure

- Disk drives are addressed as large 1-dimensional arrays of logical blocks, where the logical block is the smallest unit of transfer
- The 1-dimensional array of logical blocks is mapped into the sectors of the disk sequentially
  - Each logical block is one or more sectors
  - Sector 0 is the first sector of the first track on the outermost cylinder
  - Mapping proceeds in order through that track, then the rest of the tracks in that cylinder, and then through the rest of the cylinders from outermost to innermost.
  - Thus the *first sector* lies on the outermost track of the first platter surface, whereas the last sector lies on the innermost track of the last platter surface.



Seek time Vs Latency

Positioning time (random-access time) is time to move disk arm to desired cylinder (seek time) plus the time for desired sector to rotate under the disk head (rotational latency)



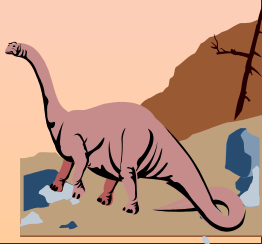
Moving from sector C to sector D

- Locate desired cylinder(seek)
- Locate desired sector (latency)

Translation from Logical Block No. to Physical Block No.

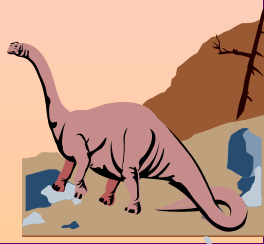
- Physical Disk Block ->(drive, cylinder, track, sector)
- Logical Disk Block is serial no of a block considering disk as a linear array of disk blocks
- File's Logical Block is a serial number of a file block starting with 0 as a first block
- File-system Organization Module performs the translation:

File's Logical Block -> Disk Logical Block -> Physical Disk Block



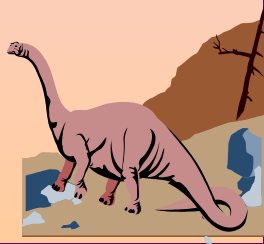
Overview of Mass Storage Structure (Cont)

- Magnetic tape
  - Was early secondary-storage medium
  - Relatively permanent and holds large quantities of data
  - Access time slow
  - Serial access : ~1000 times slower than disk
  - Mainly used for backup, storage of infrequently-used data, transfer medium between systems
  - Kept in spool and wound or rewound past read-write head
  - Once data under head, transfer rates comparable to disk
  - 20-200GB typical storage



File Concept

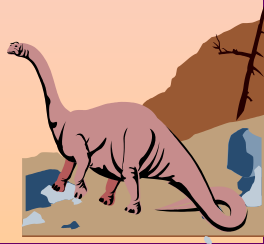
- Contiguous logical address space mapped by operating system onto physical address space on the disk which may be discontinuous.
- It is a named collection of related information that is recorded on secondary storage.
- Operating System provides the uniform view of information storage.
- In general, a file is a sequence of bits,bytes,lines, or records whose meaning is defined by the file's creator and user.
- Types:
  - Data
    - ✓ numeric
    - ✓ character
    - ✓ binary
  - Program



File Structure

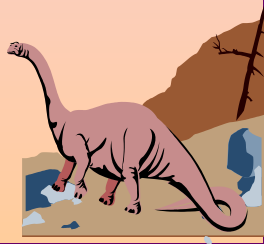
- Information in the file is defined by its creator.
- A file has certain structure according to its type.
- Who decides the file structure:
  - Operating system
  - Application Programs
- File types also may indicate the internal structure of the file.
- It is useful for an operating system to support structures that will be used frequently, and that will save the programmer's substantial effort.
- Two few structures-make programming inconvenient

  Two many structures-the resulting size OS is cumbersome



File structure Support in UNIX

- Considers each file to be a sequence of 8-bit bytes, no interpretation of these bits is done
- Each application program must include its own code to interpret an input file

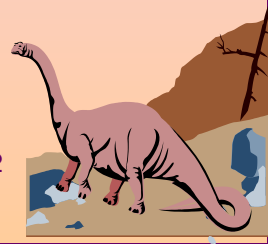


File Types – Name, Extension

file type	usual extension	function
executable	exe, com, bin or none	read 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, rrf, doc	various word-processor formats
library	lib, a, so, dll, mpeg, mov, rm	libraries of routines for programmers
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, rm	binary file containing audio or A/V information

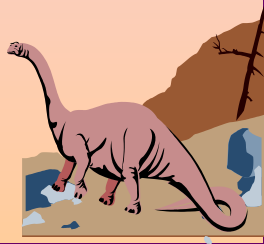
File types Contd…

- MSDOS supports few file extensions like .bat, .com, .exe, .obj
- UNIX also allows file name extension hints, but not enforced. They simply aid the users in determining the type of contents of the file.
- In Windows,
  - file extensions are not mandatory, but they are commonly used and recommended for proper file organization.
  - file extensions are typically three or four characters long and follow the format ".extension". For example, ".txt", ".docx", ".jpg".



File System

- A file system is a method or structure used by operating systems to organize, store, and retrieve files on a storage device such as a hard disk drive or solid-state drive.
- It provides a way to manage and control access to files, directories, and other data stored on the storage medium.
- File systems define the rules and structures for naming, organizing, and storing files, as well as managing metadata associated with the files.



File System Examples

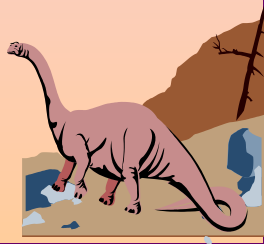
Operating System	File Systems Supported
Windows	NTFS, FAT32, exFAT, ReFS (Windows 10 and above)
macOS	HFS+, APFS
Linux	ext4, ext3, ext2, XFS, Btrfs, JFS, ZFS
Android	ext4, exFAT, F2FS, NTFS
iOS	APFS
FreeBSD	UFS, ZFS, FAT32
Solaris	ZFS, UFS, NFS, FAT32
Chrome OS	ext4, FAT32
Unix	UFS, ZFS, NFS

Internal file structure

- All disk I/O is performed in units of one disk block (cluster)
- All clusters are of same size for a particular file system(say 1024 bytes)
- Logical record size may vary(made of one or more physical blocks)
- packing a number of logical records into physical blocks is a common solution
- Internal fragmentation occurs

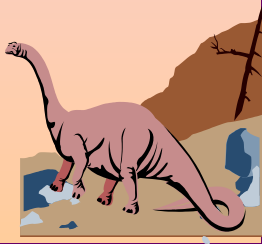
Ex. Calculate internal fragmentation for a file of 1924 bytes stored on it. The system is having 1024 bytes disk block(cluster)size.

- All file systems suffer from internal fragmentation
- The larger the disk block size, the greater the internal fragmentation.



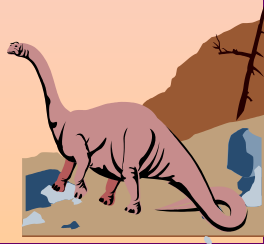
File Attributes

- **Name** – only information kept in human-readable form.
- **Type** – needed for systems that support different types.
- **Location** – pointer to file location on device.
- **Size** – current file size.
- Name of the creator
- **Protection** – controls who can do reading, writing, executing.
- **Time**, **date**, **and user identification** – data for protection, security, and usage monitoring.
- Information about files are kept in the *directory structure*, which is maintained on the disk.



File Operations

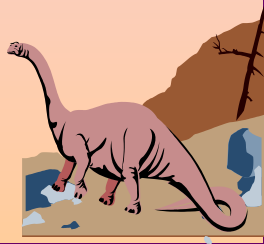
- Create
- Write
- Read
- Reposition within file – file seek
- Delete
- Truncate
- Open(F_i) – search the directory structure on disk for entry F_i, and move the content of entry to memory.
- Close (F_i) – move the content of entry F_i in memory to directory structure on disk.



File Access Methods

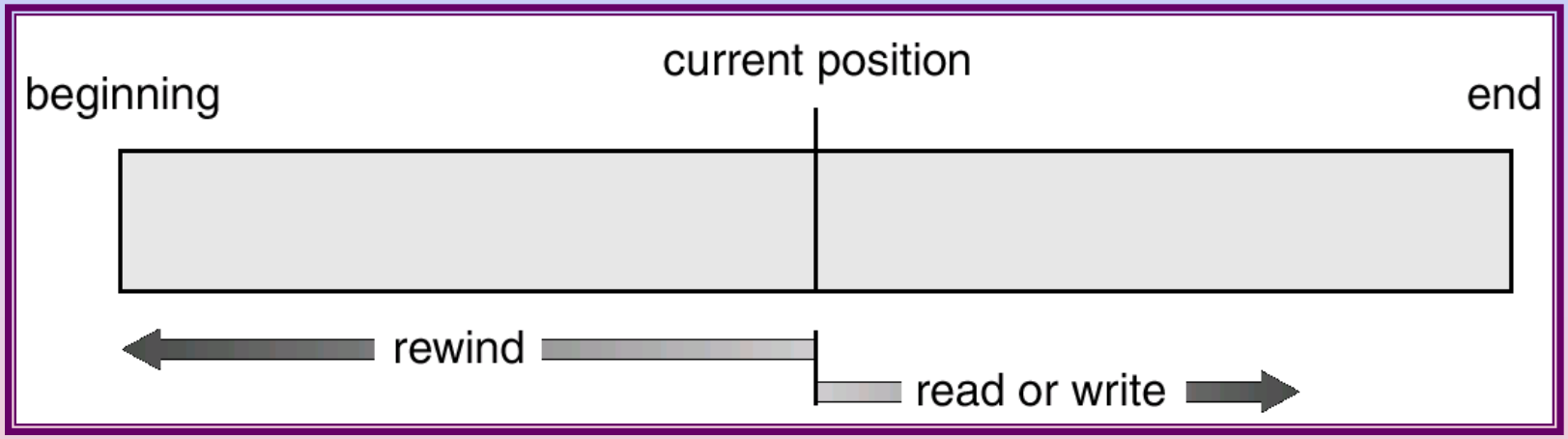
- Sequential Access
- Random Access

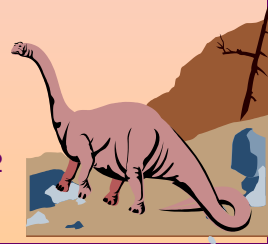
- Sequential Access
  - Based on a tape model of a file
  - Operations:
    read next
    write next
    skip n
    reset/rewind



Sequential-access File

- Modelled after magnetic tape storage





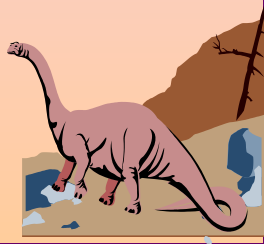
Direct Access/Random access/Hashed

- **Based on a disk model of a file**
- **A file is made of fixed length logical records**
- **Operations:**
  - *read n*
  - *write n*
  - *position to n*
  - *read next*
  - *write next*

n = relative block/record number (an index relative to the start of the file)

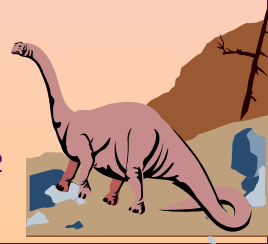
Ex. 1. An airline reservation system: A file storing all the info about all flights. Info about flight 713 may be stored in block 713 of reservation file

Ex. 2 To access an employee record in an employee DB, a hashing function on employee ID may be used to compute the block number.



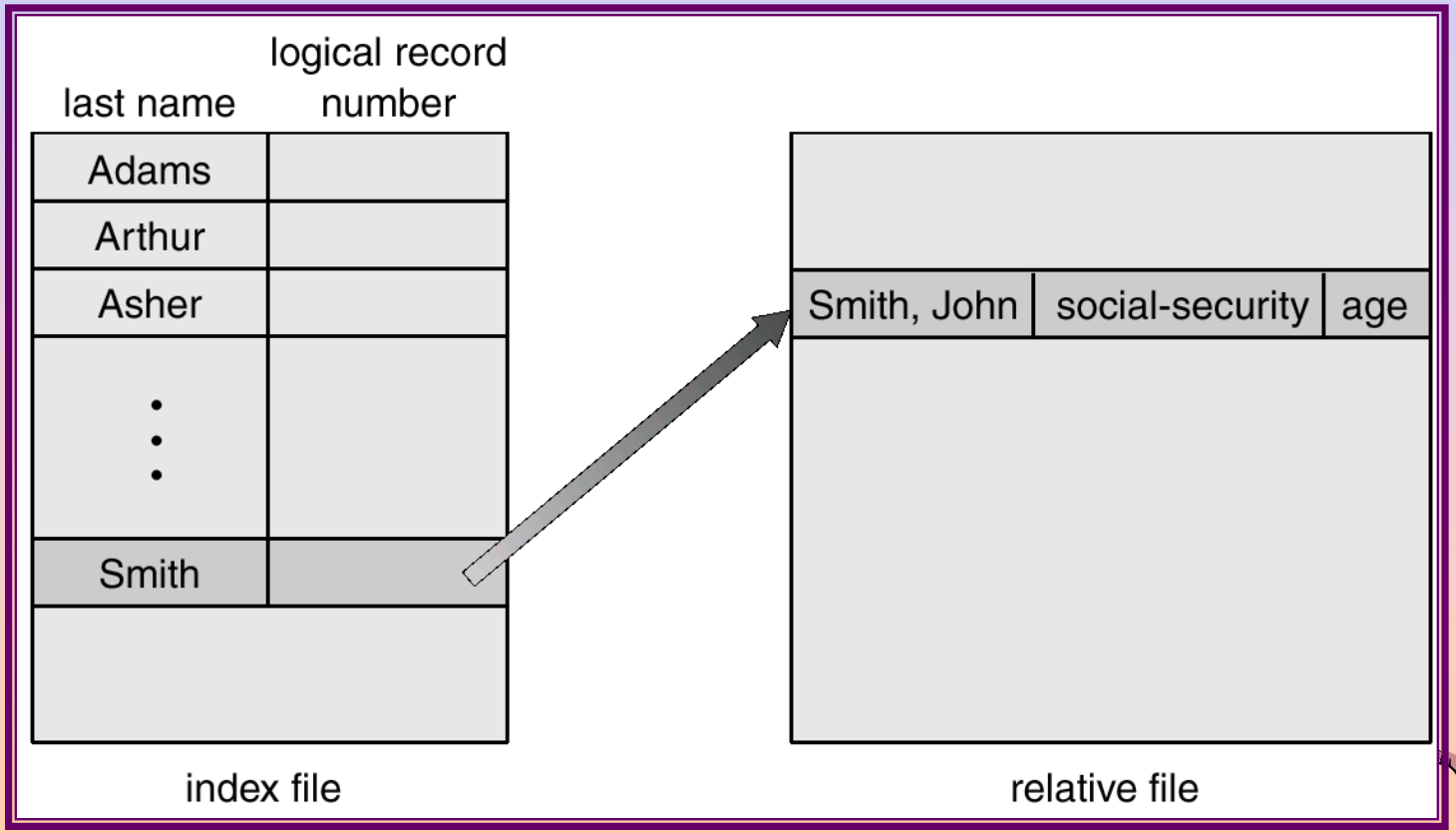
Simulation of Sequential Access on a Direct-access File

sequential access	implementation for direct access
reset	*cp* = 0;
read next	*read cp*; *cp* = *cp*+1;
write next	*write cp*; *cp* = *cp*+1;



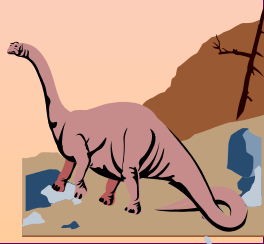
Other Access Methods
Example of Index and Relative Files

- Based on top of direct access method
- Index file based on a key field needs to constructed (sorted on key field)
- Index file is searched with help of binary search



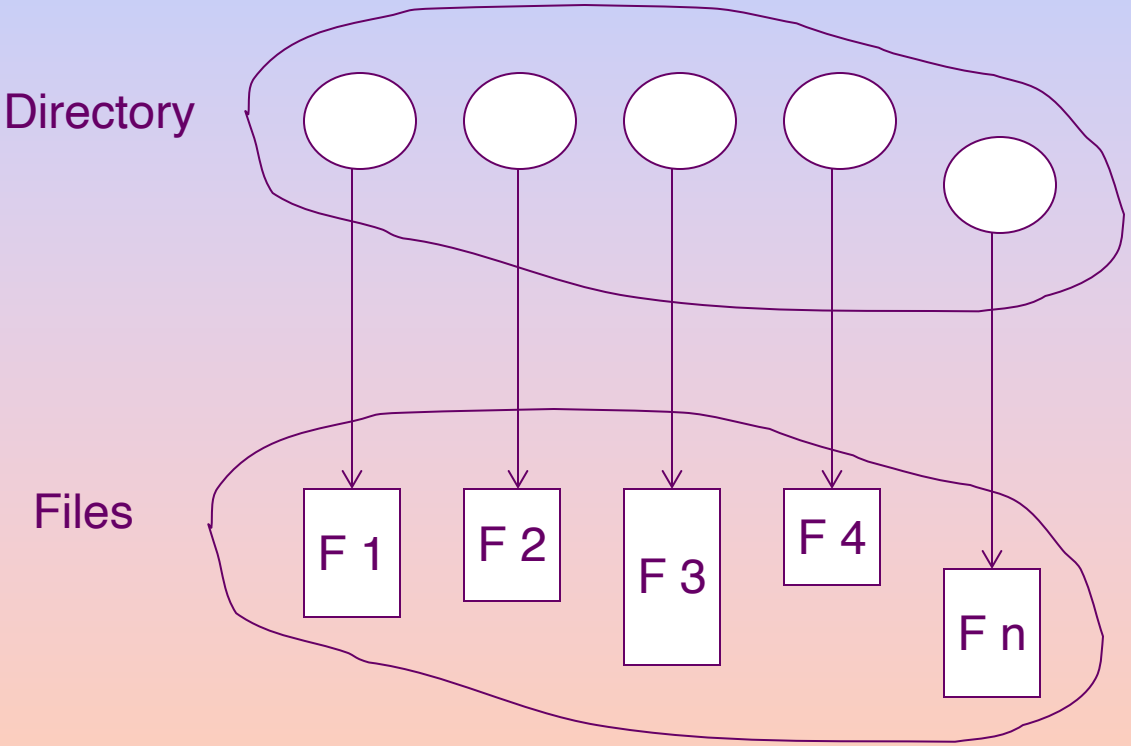
Parts of File System

- Collection of files (each storing related data)
- Directory Structure (Organizes and provides info about all the files in the system)
- Partitions (used to separate physically or logically large collections of directories)

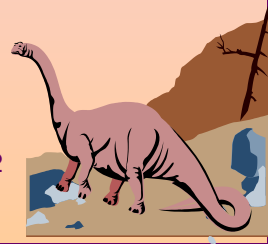


Directory Structure

- A collection of nodes containing information about all files.
- Both the directory structure and the files reside on disk.
- Backups of these two structures are kept on tapes.

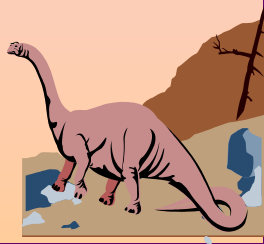


A directory has many files

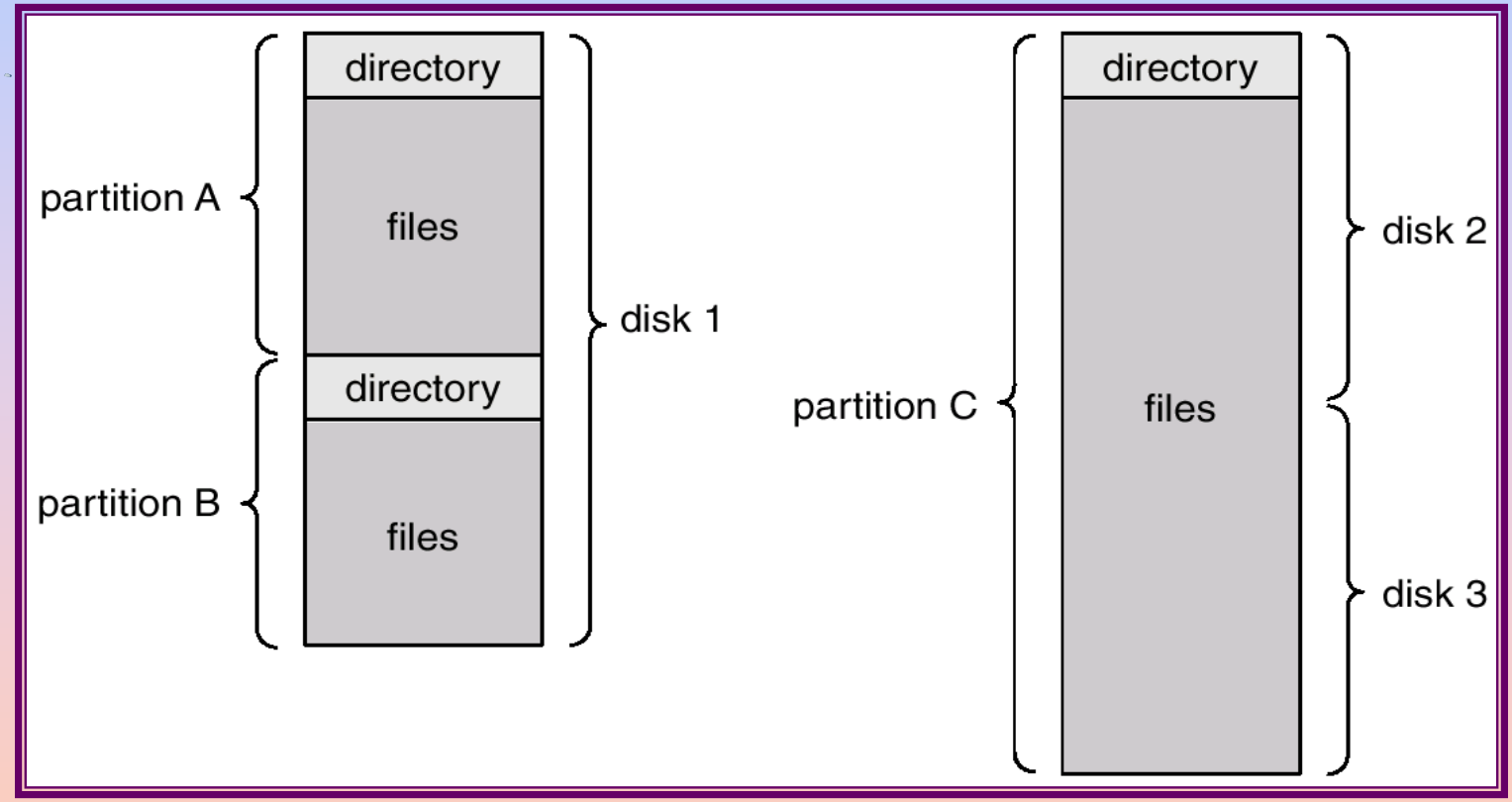


Partitioning

1. The storage device may be divided into partitions, which are logical divisions of the disk space.
2. Each partition is treated as a separate entity with its own file system and may have a specific format (e.g., FAT32, NTFS, ext4).
3. A partition is a logical division or section of a physical disk drive.
4. The terms "volume" and "partition" are often used interchangeably



A Typical File-system Organization

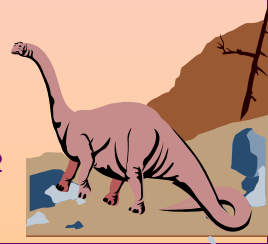


Example

- This example concerns a Windows system with two physical hard disks.
- The first hard disk has two partitions, the second has only one.
- The first partition of the first hard disk contains the operating system.

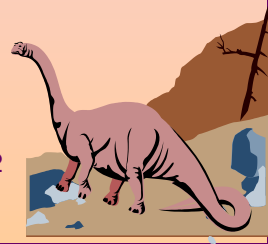
Physical disk	Partition	File System	Drive letter
Hard Disk 1	Partition 1	NTFS	C:
	Partition 2	FAT32	D:
Hard Disk 2	Partition 1	FAT32	E:

- "C:", "D:", and "E:" are volumes.
- Hard Disk 1 and Hard Disk 2 are physical disks.
- Any of these can be called a "drive".



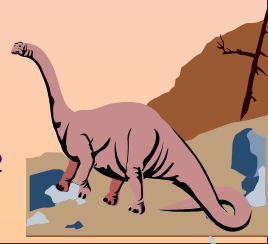
Information in a Disk Directory / Volume Table of Contents

- Disk Directory / VTOC is a data structure within a file system that stores information about files and directories on a disk or storage volume.
- The contents of a disk directory typically include the following entry about each file :
  - Name
  - Type
  - Address
  - Current length
  - Maximum length
  - Date last accessed (for archival)
  - Date last updated (for dump)
  - Owner ID (who pays)
  - Protection information (discuss later)



Operations Performed on Directory

- Search for a file
- Create a file
- Delete a file
- List a directory
- Rename a file
- Traverse the file system

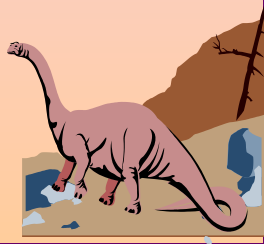


Organize the Directory (Logically)

- **Efficiency** – locating a file quickly.
- **Naming** – convenient to users.
  - Two users can have same name for different files.
  - The same file can have several different names.
- **Sharing-** same file may be shared by different users
- **Grouping** – logical grouping of files by properties, (e.g., all Java programs, all games, …)

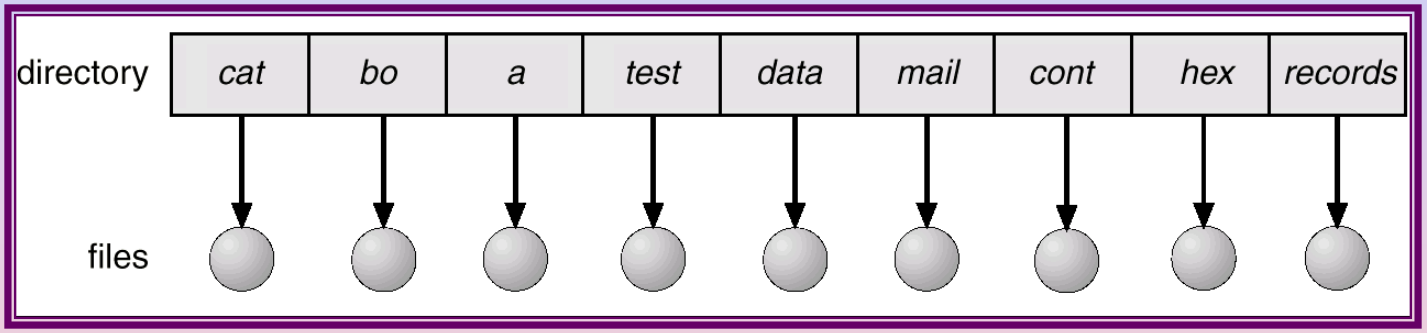
Common Schemes of Directory Structure Organization

1. Single Level
2. Two-level
3. Tree Structured
4. Acyclic –Graph
5. General Graph



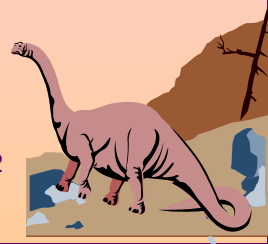
Single-Level Directory

- A single directory for all users.



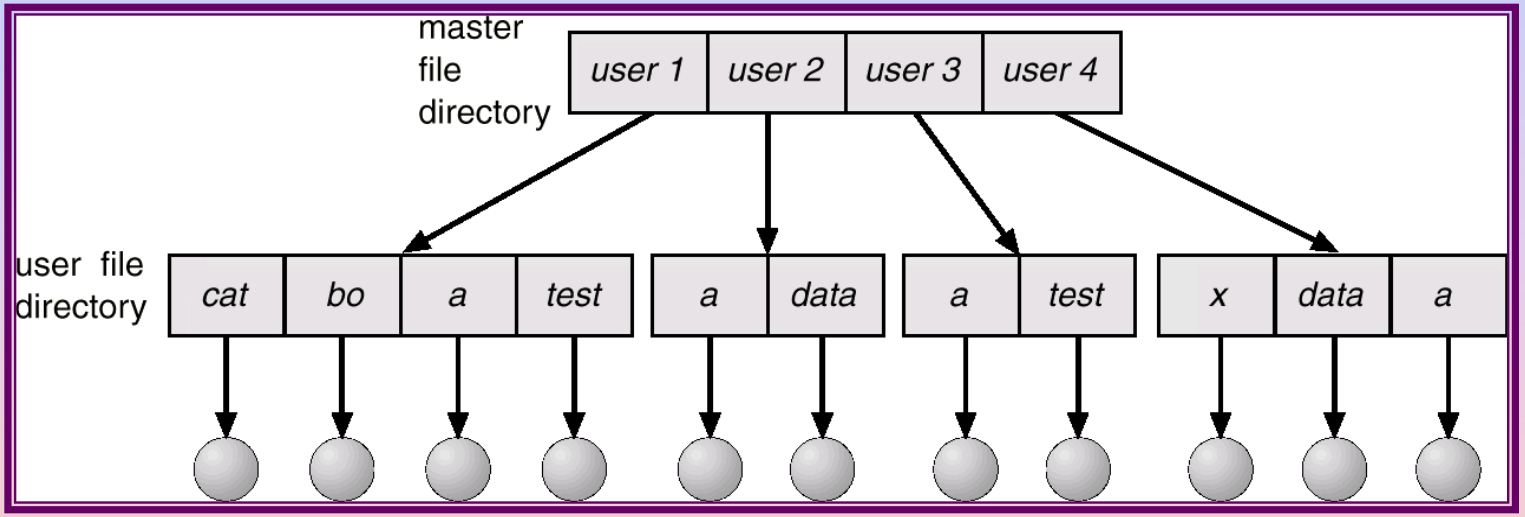
Naming problem

Grouping problem



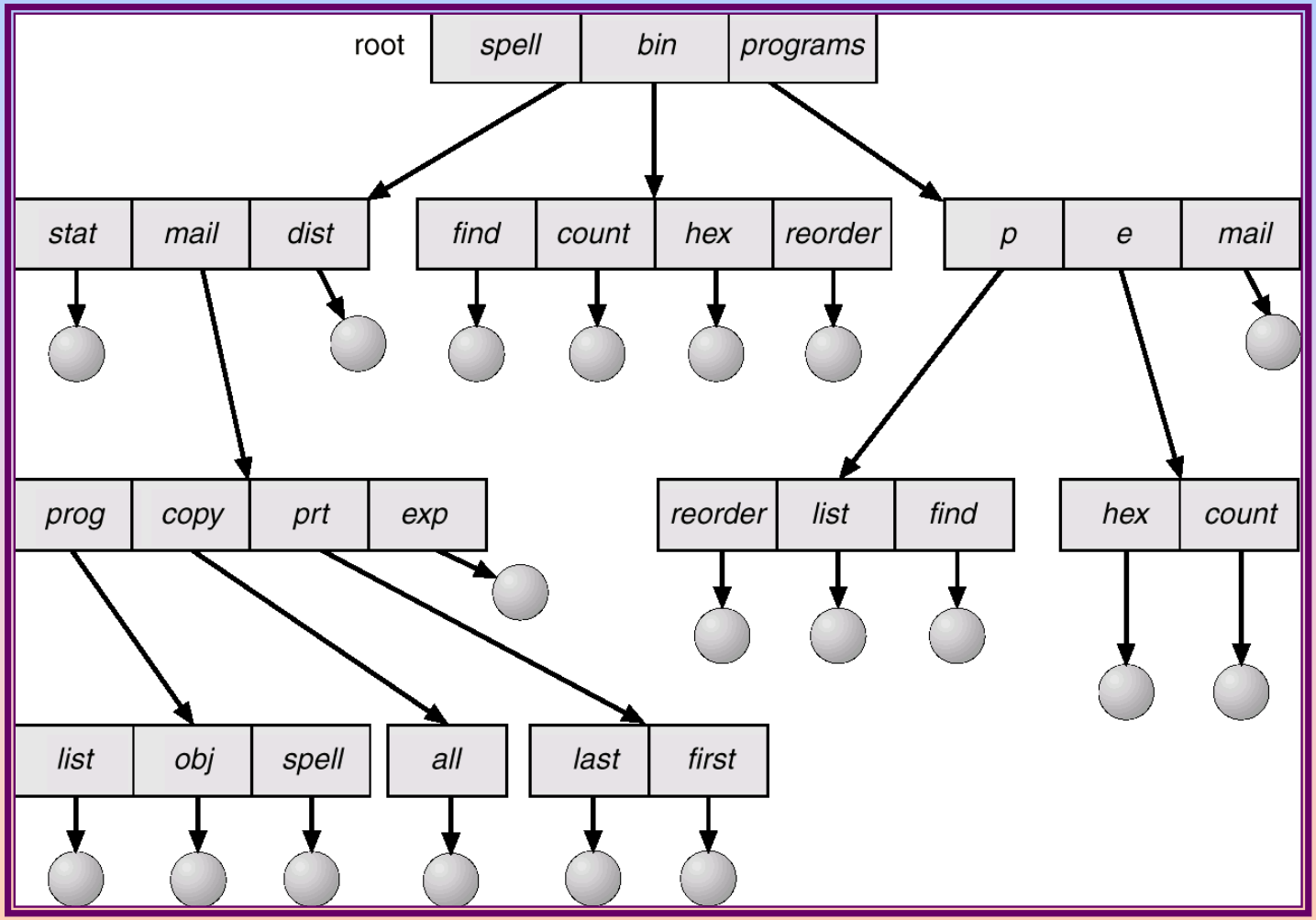
Two-Level Directory

- Separate directory for each user.



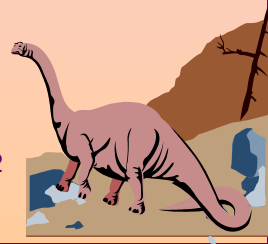
- Path name
- Can have the same file name for different user
- Efficient searching
- No grouping capability

Tree-Structured Directories



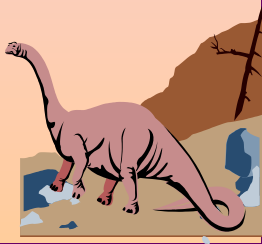
Tree-Structured Directories (Cont.)

- Efficient searching
- Grouping Capability
- Current directory (working directory)
  - **cd** /spell/mail/prog
  - **type** list



PATH

- an **absolute path** refers to a specific location in a file system ***relative to the root directory***,
- whereas a **relative path** points to a specific location in a file system ***relative to the current directory*** you are working on
- Every operation on file/subdirectory will be done on the current directory contents, unless otherwise specified explicitly using the absolute path for that file/subdirectory



Tree-Structured Directories (Cont.)

- Creating a new file is done in current directory.
- Delete a file

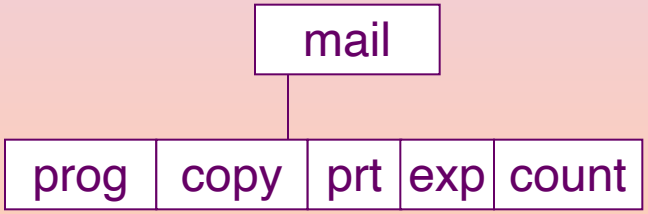
  rm <file-name>
- Creating a new subdirectory is done in current directory.

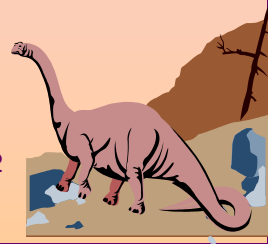
  mkdir <dir-name>

  Example: if current directory is **/mail then the command**

  mkdir count

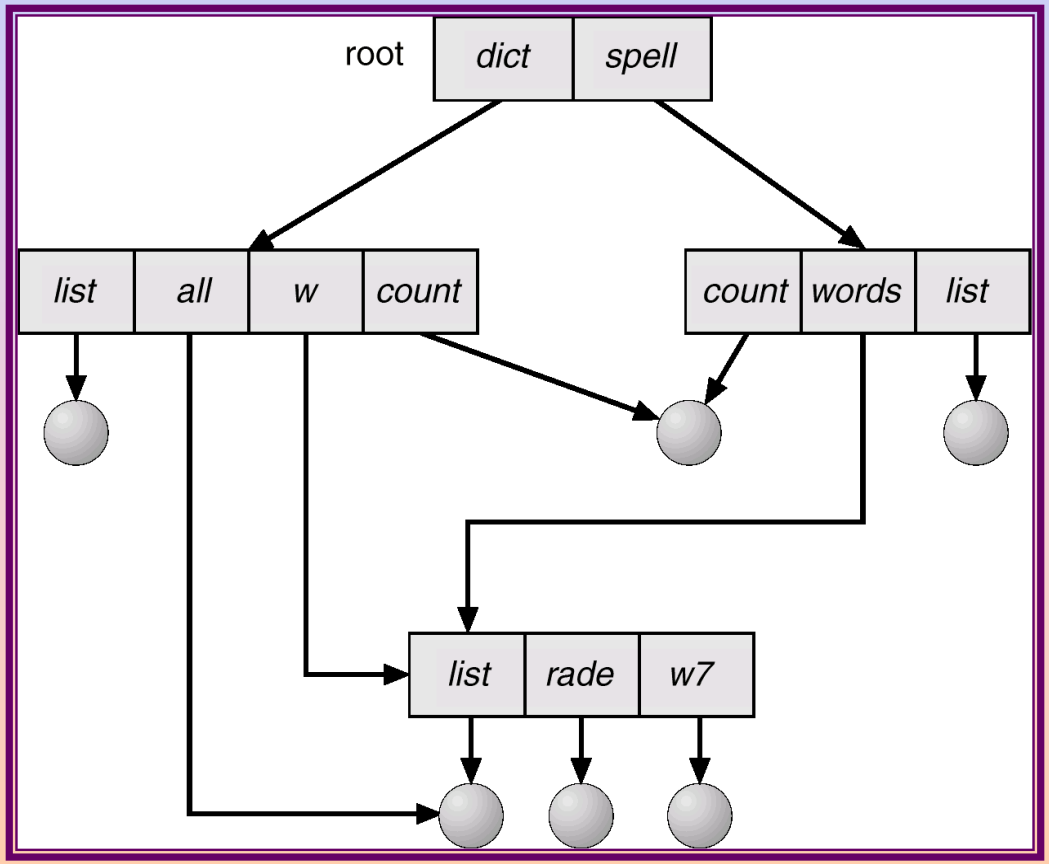
Will result in the count subdirectory as follows





Acyclic-Graph Directories

- Acyclic graph is directed graph that does not contain any cycles
- Have shared subdirectories and files.

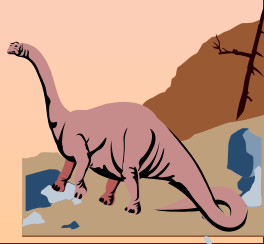


Ex. Give the two absolute paths for the following shared files /subdirectory:

1. count
2. list

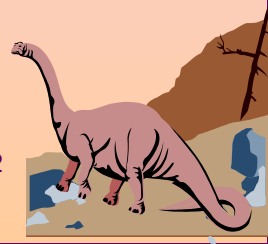
Acyclic-Graph Directories (Cont.)

- Two different names (aliasing)
- If *dict* deletes *list* $\Rightarrow$ dangling pointer.
  Solutions:
  - Backpointers, so we can delete all pointers. Variable size records a problem.
  - Entry-hold-count solution.



File Protection

- File owner/creator should be able to control:
  - what can be done
  - by whom
- Types of access
  - Read
  - Write
  - Execute
  - Append
  - Delete
  - List

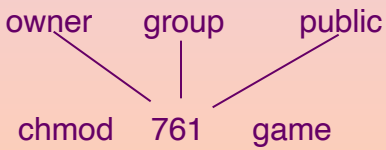


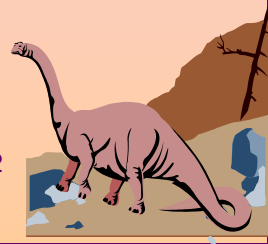
Access Lists and Groups

- Mode of access: read, write, execute
- Three classes of users

			RWX
a) **owner access**	7	⇒	1 1 1
b) **group access**	6	⇒	RWX 1 1 0
c) **public access**	1	⇒	RWX 0 0 1

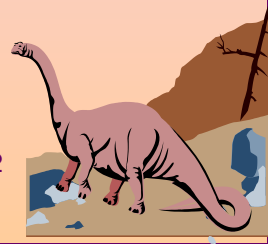
- Ask manager to create a group (unique name), say G, and add some users to the group.
- For a particular file (say *game*) or subdirectory, define an appropriate access.





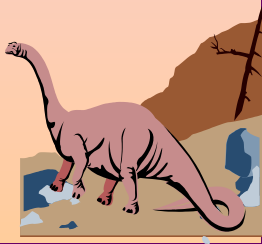
Implementation Issues

- File System Structure
- File System Implementation
- Directory Implementation
- Allocation Methods
- Free-Space Management
- Efficiency and Performance

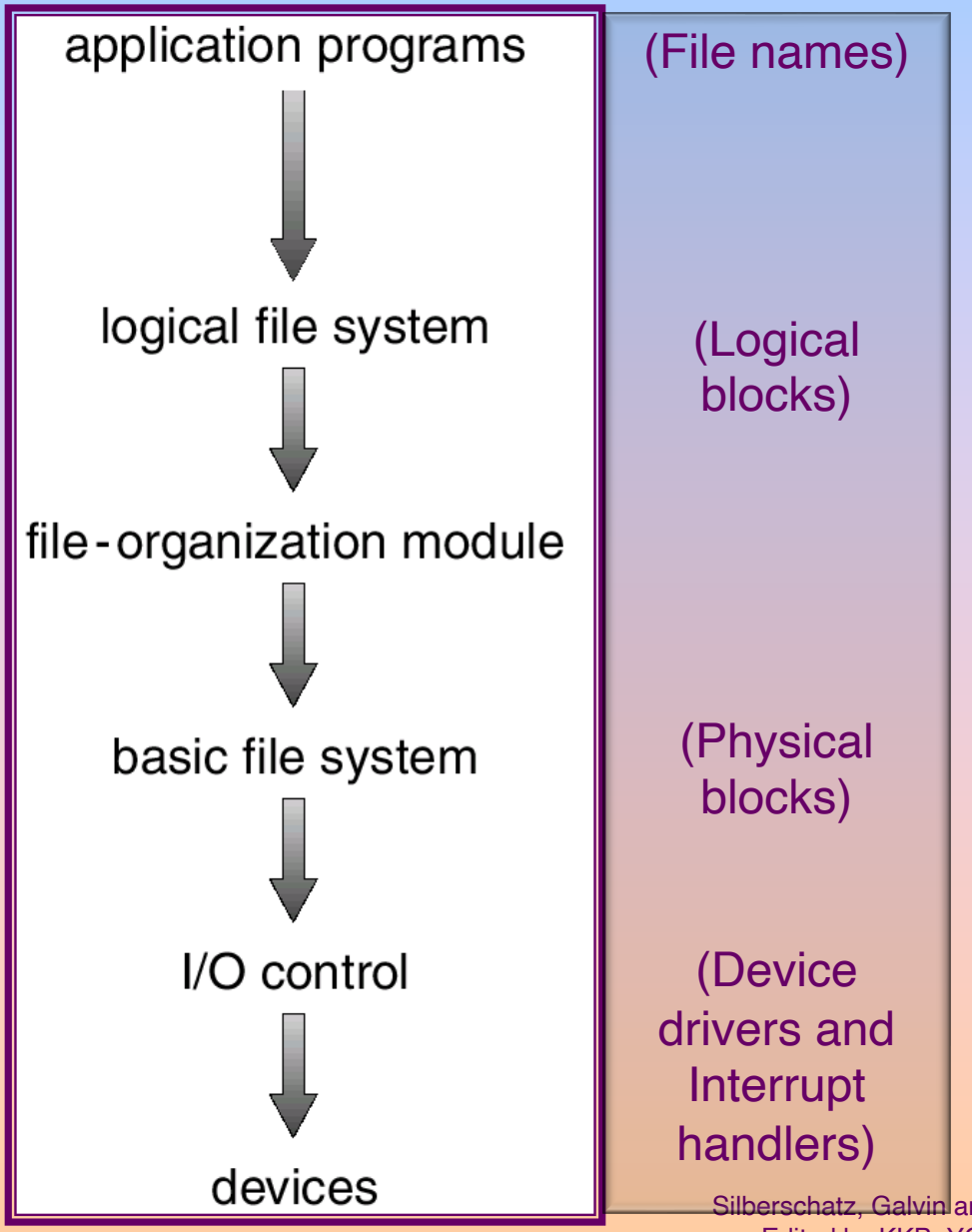


File-System Structure

- File structure
  - Logical storage unit
  - Collection of related information
- File system organized into layers
- File system resides on secondary storage (disks)
  - Provides efficient and convenient access to disk by allowing data to be stored, located retrieved easily
- File control block – storage structure consisting of information about a file
- Device driver controls the physical device



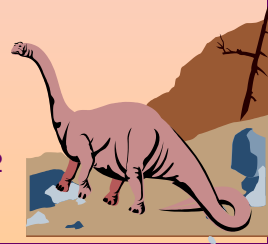
Layered File System



A Typical File Control Block (for each file)

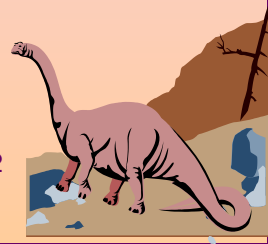
- A **File Control Block** (**FCB**) is a **file** system structure in which the state of an open **file** is maintained.
- A **FCB** is managed by the operating system, but it resides in the memory of the program that uses the **file**, not in operating system memory.

file permissions
file dates (create, access, write)
file owner, group, ACL
file size
file data blocks or pointers to file data blocks



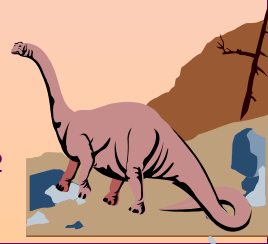
File Descriptor Table (FDT) vs File Control Block(FCB)

- FDT one for each process whereas FCB is one for each file
- the File Descriptor Table is a data structure specific to each process, maintaining information about the files opened by that process.
- On the other hand, File Control Block (FCB) is a data structure specific to each file, holding metadata and facilitating file operations for that file.
- The FCB contains metadata and information about the file, such as its name, size, location, permissions, timestamps, and other attributes.
- The FDT helps the operating system manage open files within a process, while the FCB allows the operating system to manage and control access to individual files.



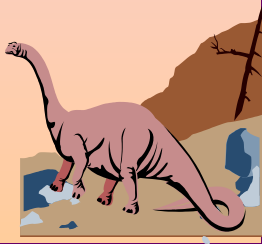
Allocation Methods

- An allocation method refers to how disk blocks are allocated for files:
- Contiguous allocation
- Linked allocation
- Indexed allocation

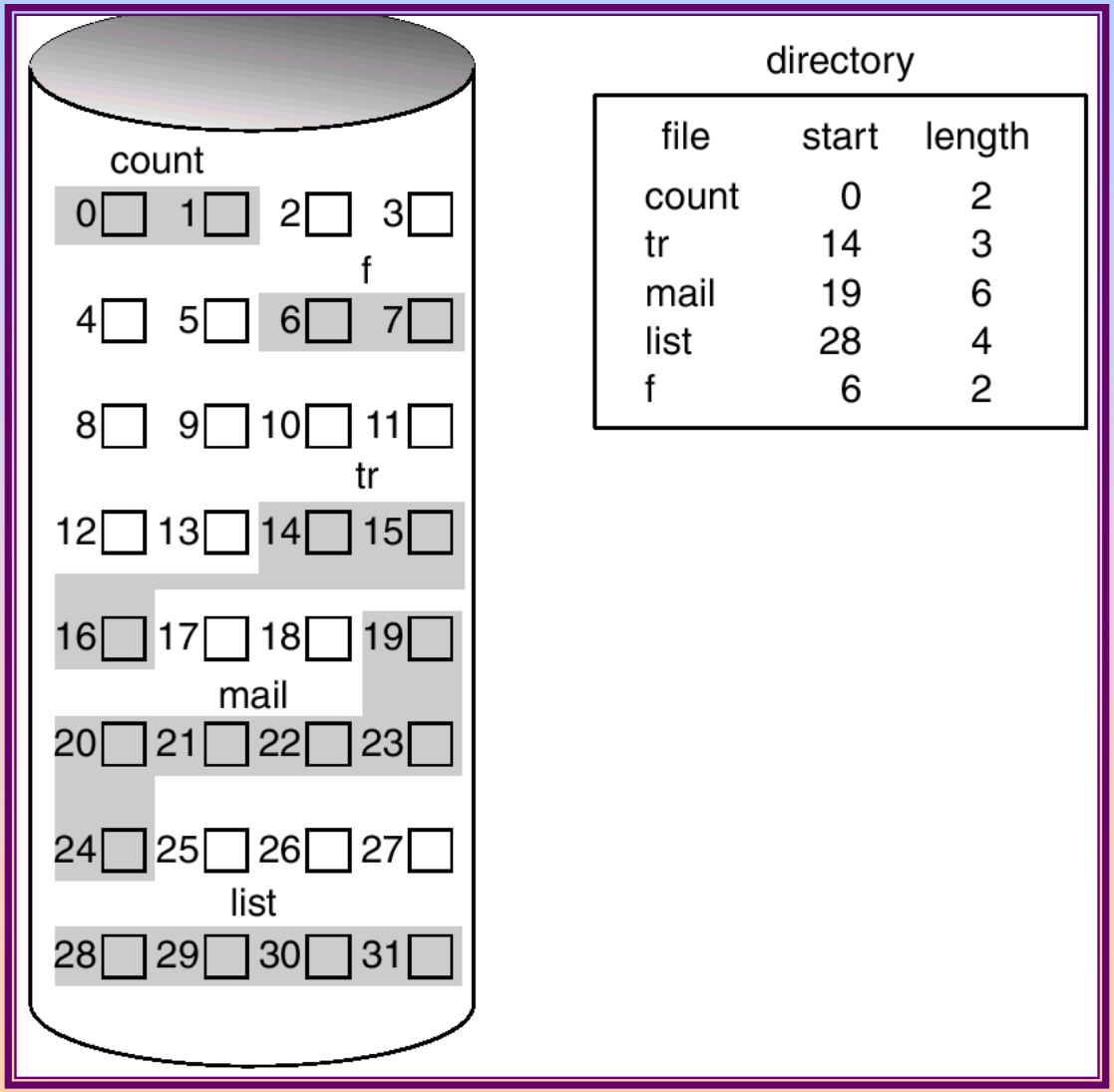


Contiguous Allocation

- Each file occupies a set of contiguous blocks on the disk.
- Simple – only starting location (block #) and length (number of blocks) are required.
- Supports Sequential and Random access.
- Wasteful of space (External Fragmentation).
- Files cannot grow.



Contiguous Allocation of Disk Space

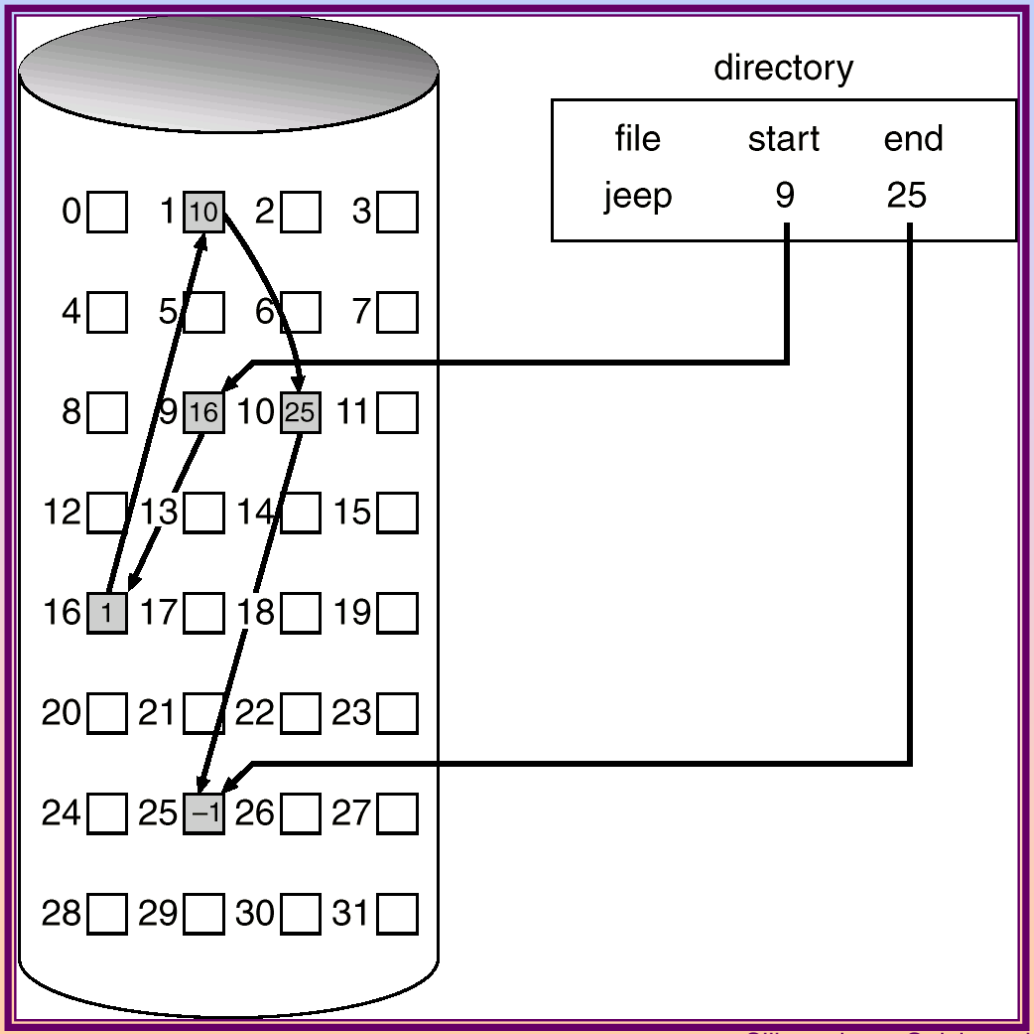


directory

file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Linked Allocation

- Each file is a linked list of disk blocks: blocks may be scattered anywhere on the disk.



Linked Allocation (Cont.)

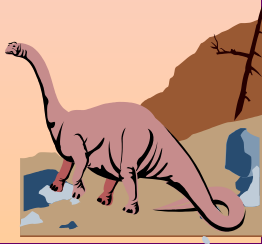
- Simple – need only starting address
- Free-space management system – no waste of space
- No random access
- Mapping

LogicalAddress / BLOCK_SIZE → Quotient Q, Remainder R

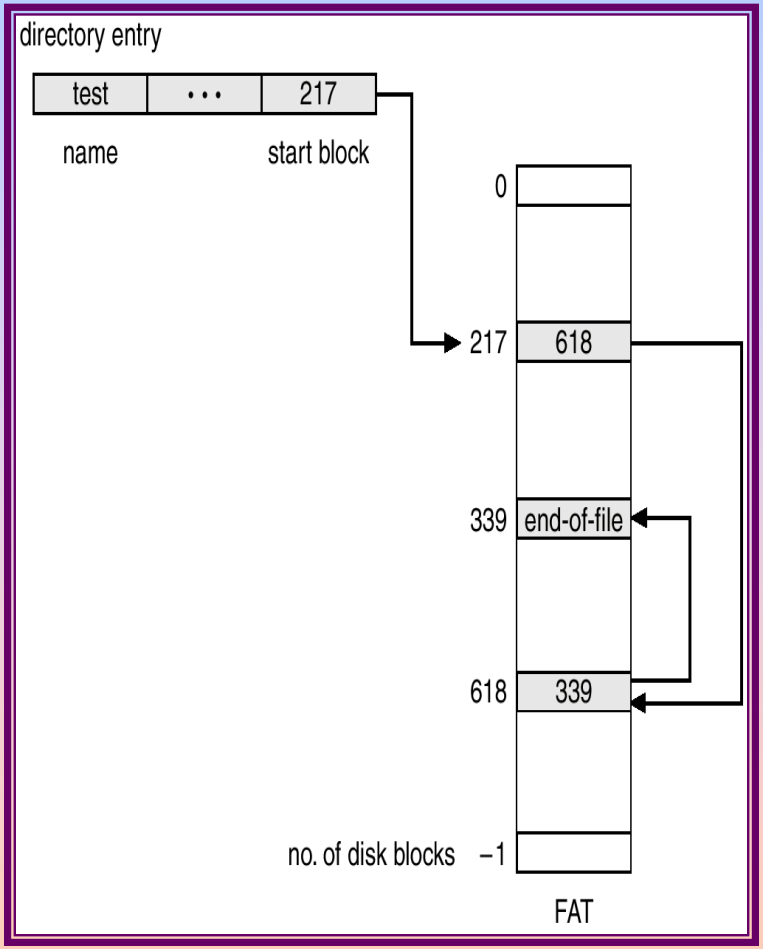
Block to be accessed = Q^{th} block in the linked chain of blocks
Displacement into block = R

E.g. Find the block number and displacement for the Logical address 1234 in a file in the linked allocation with a block size of 512 bytes.
ANS: (Block No.,offset) =(1234//512, (1234%512))= (2,210)

- File-allocation table (FAT) – disk-space allocation used by MS-DOS and OS/2 uses linked allocation



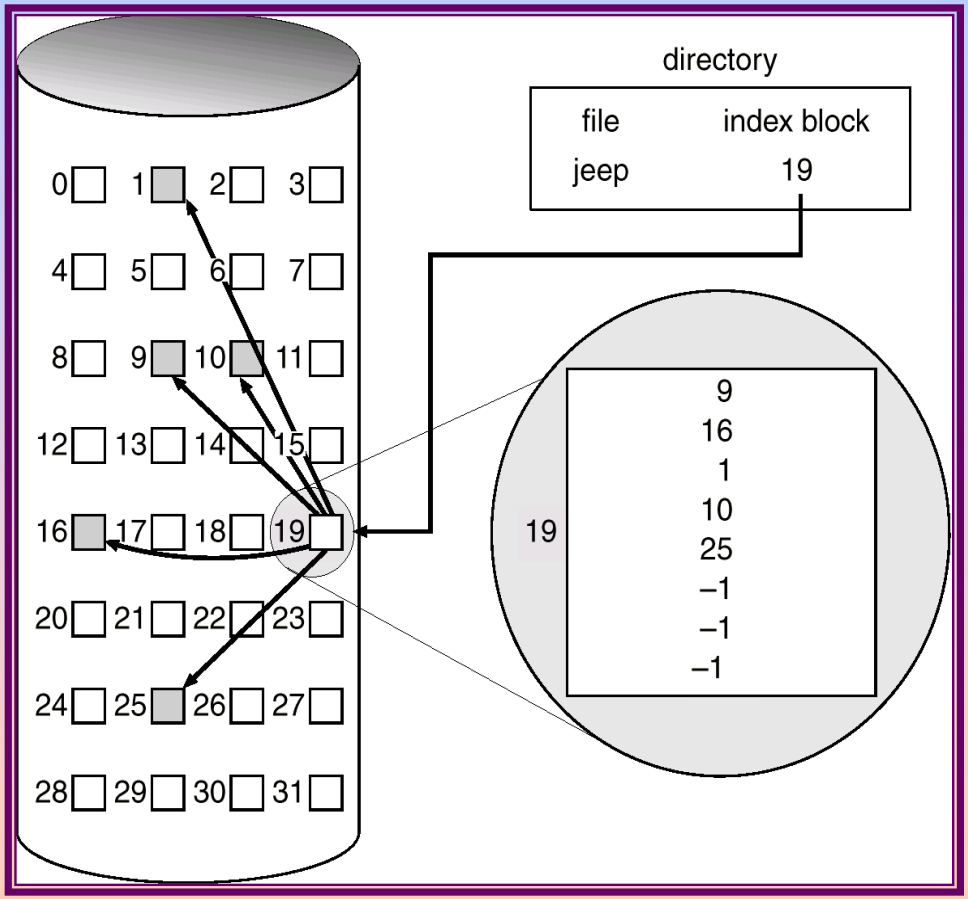
File-Allocation Table



- This is an implementation of linked allocation scheme used by MSDOS and OS2 operating systems.
- A section of the disk at the beginning of each partition is set aside to contain FAT.
- Directory entry just provides starting disk block number allocated to the file.
- The table has one entry for each disk block.
- Free Space management: Entry 0 indicated free block
- EOF indicates end of file.
- Non-zero numbers indicate number of the next disk block in sequence, allocated to the file.
- Finding a free block is easy. No separate free space management system required

Number of FAT entries = No. of disk blocks

Indexed Allocation

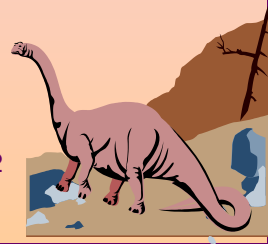


- Has separate index block for each file
- Brings all pointers together into the *index block*
- Random access is possible here
- No External Fragmentation

Best Allocation Method

- Depends on file size and type of access required
- Contiguous allocation is best for files that are small and accessed sequentially.
- Linked allocation is best for large files accessed sequentially.
- Indexed allocation is best for large, randomly accessed files.

Allocation Method	Supported File Access Methods	File Size preferred
Contiguous	Sequential, direct(random)	small
Linked	Sequential	large
Indexed	Sequential, Direct(random)	large

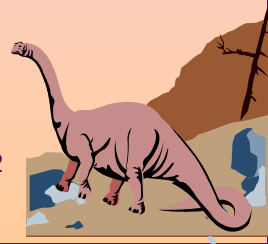


i-Node (Index node)

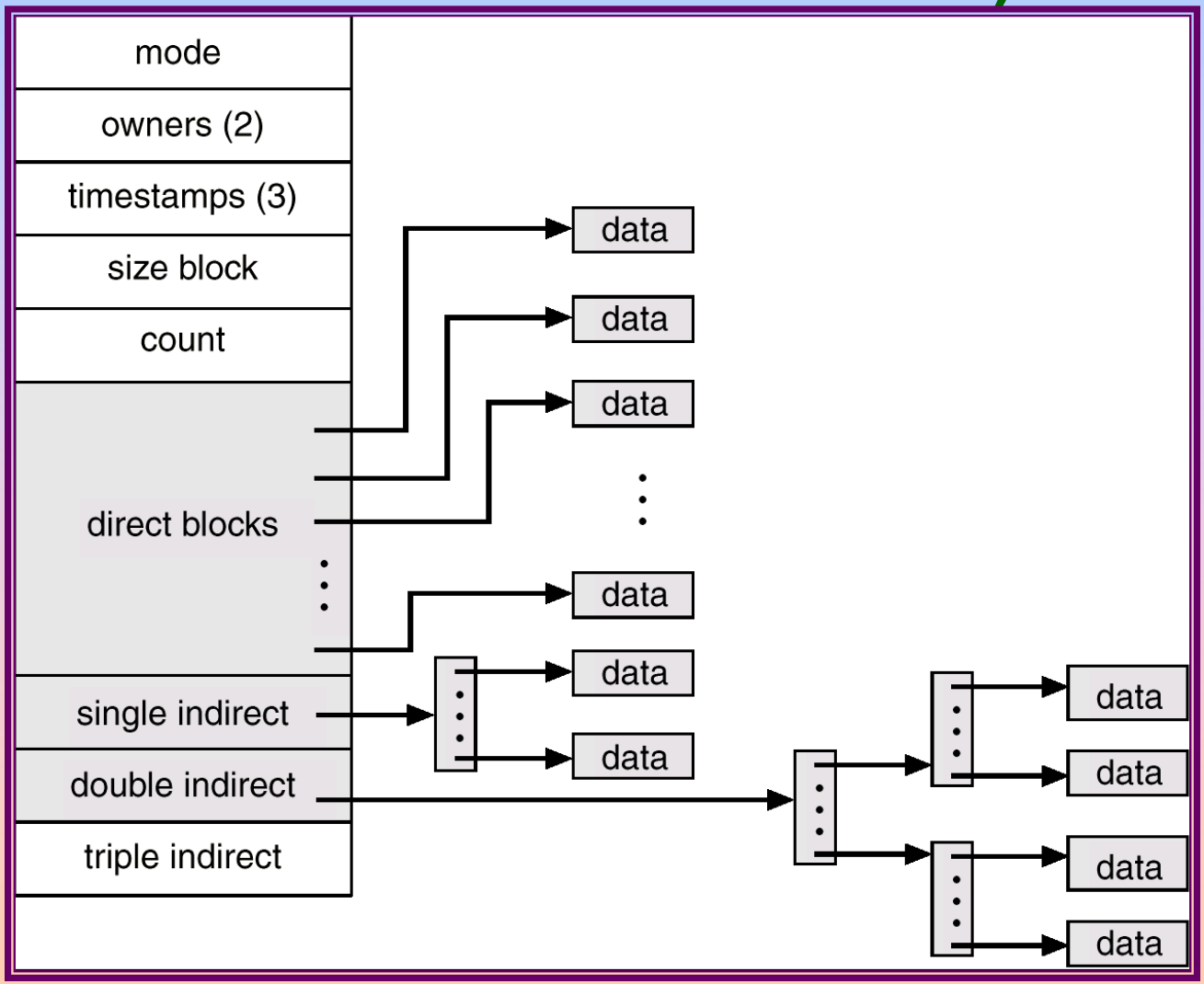
- It is a data structure on a traditional Unix-style file system such as UFS

- An i-node stores all the information about regular file, directory, and other file system objects except its data and name.

- It also stores pointers to data blocks using multilevel indexing scheme.



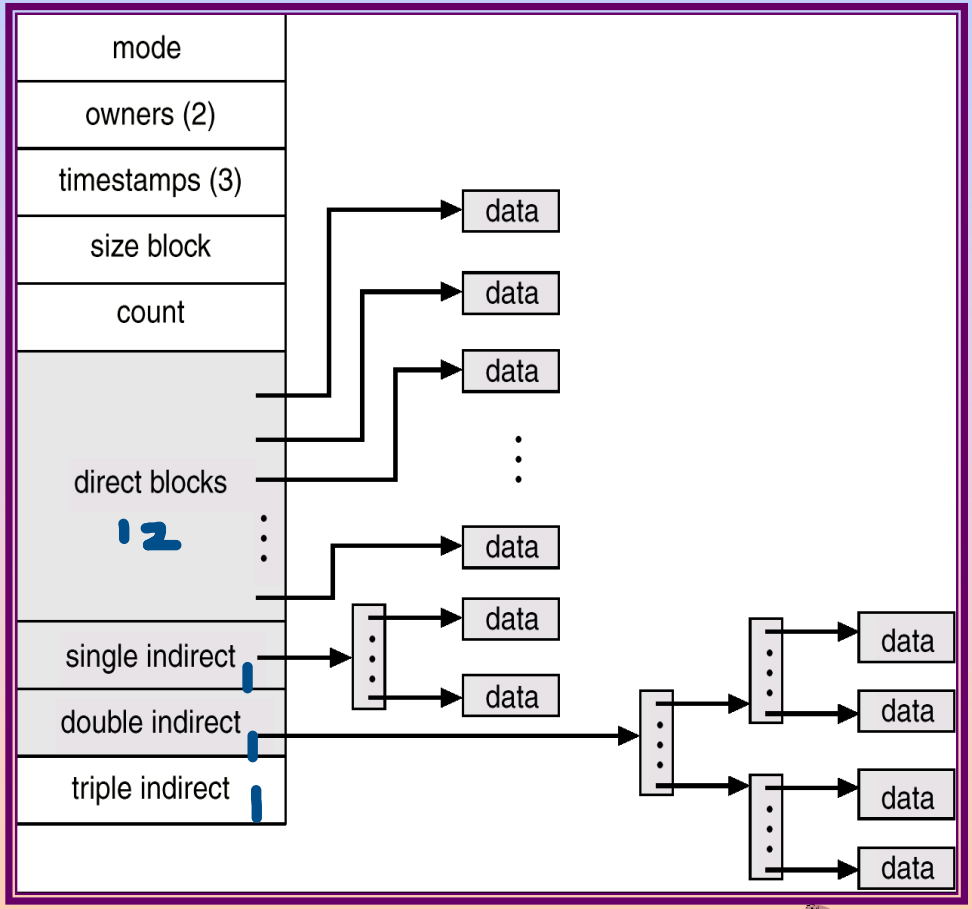
Combined Scheme: UNIX i-node (similar to FCB in MS Windows)



Example: Calculating max file size using i-node scheme

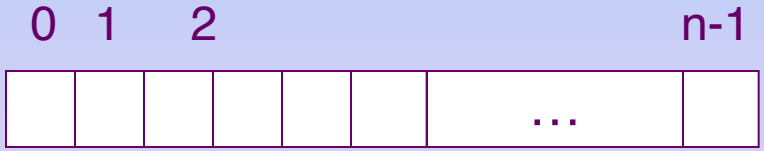
Calculate the maximum size of a file for a file system with multilevel indexing scheme. Each block is of size 1K and size of a block pointer is 4 bytes. Suppose each block has 15 block pointers; with 12 direct pointers, and one pointer each for single, double and triple indirect blocks.

Ans. Number of pointers per block = block size/pointer size=1024/4=256
Max. File Size= (12+256+256x256+256x256x256)KB >16 GB



Free-Space Management

- **Scheme-1**: **Bit vector** (*n* blocks)



$$bit[i] = \begin{cases} 0 \Rightarrow block[i] \text{ free} \\ 1 \Rightarrow block[i] \text{ occupied} \end{cases}$$

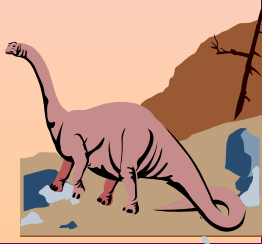
- e.g. Find the space required in KB to store a bit vector for a 6.4 GB disk with 1 KB blocks.

  Ans. Number of disk blocks of 1K size =(6.4*1024*1024)

  Each block requires one bit in the bit vector

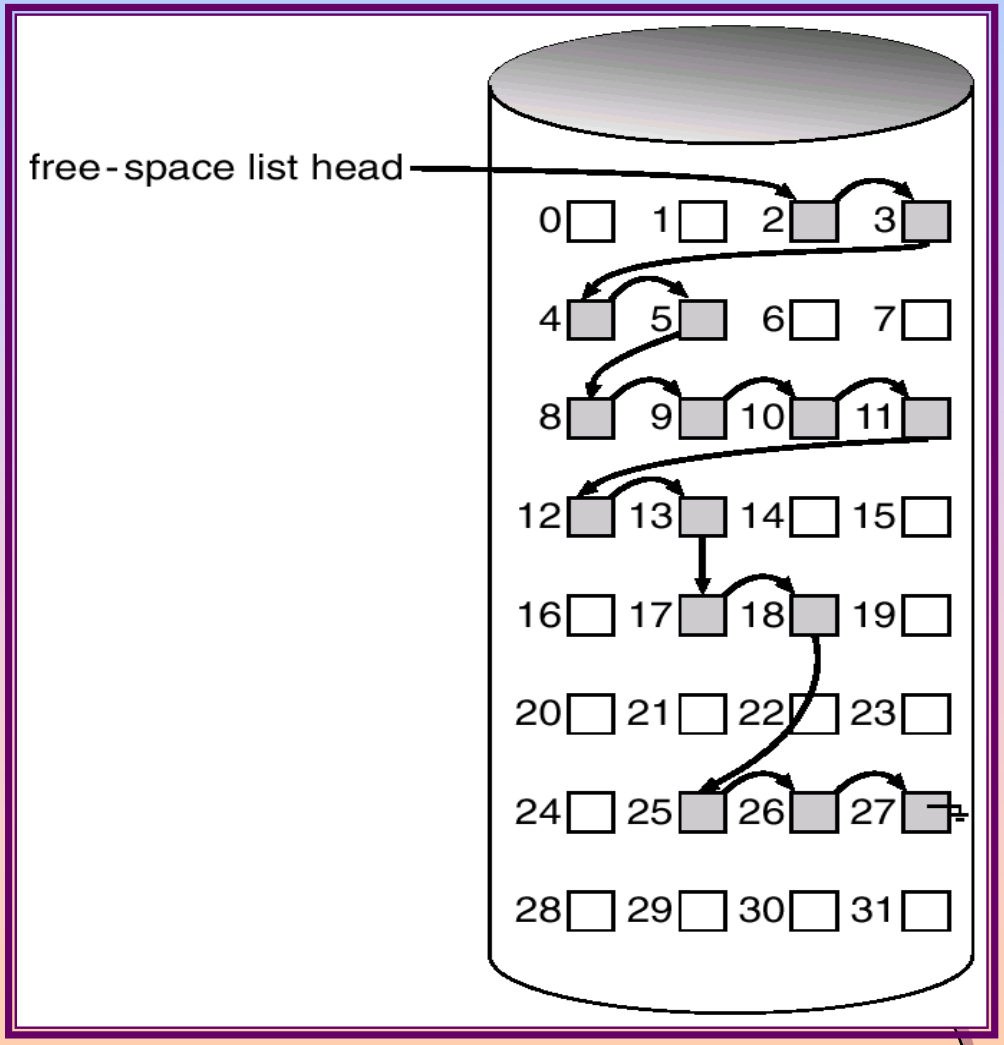
  Space required = 6.4*1024*1024 /8 bytes= (819 KB))

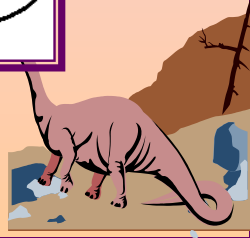
- Linked allocation does not need separate scheme for FSM



Free Space Management…

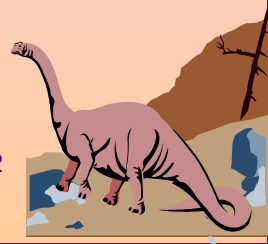
Scheme-2: Linked Free Space





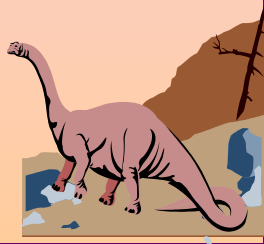
Disk Scheduling

- Disk Structure
- Disk Scheduling
- Disk Management
- Selecting a Disk-Scheduling Algorithm



Disk Scheduling Algorithms

- The operating system is responsible for using hardware efficiently — for the disk drives, this means having a fast access time and disk bandwidth.
- Access time has two major components
  - *Seek time* is the time for the disk to move the heads to the cylinder containing the desired sector.
  - *Rotational latency* is the additional time waiting for the disk to rotate the desired sector to the disk head.
- **Disk scheduling algorithms** minimizes seek time ; given the set of requests containing cylinder numbers.
- Seek time is proportional to seek distance
- **Disk bandwidth** is the total number of bytes transferred, divided by the total time between the first request for service and the completion of the last transfer.

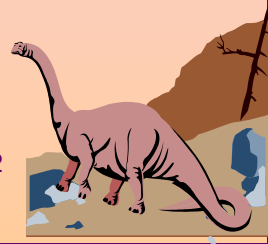


Disk Scheduling (Cont.)

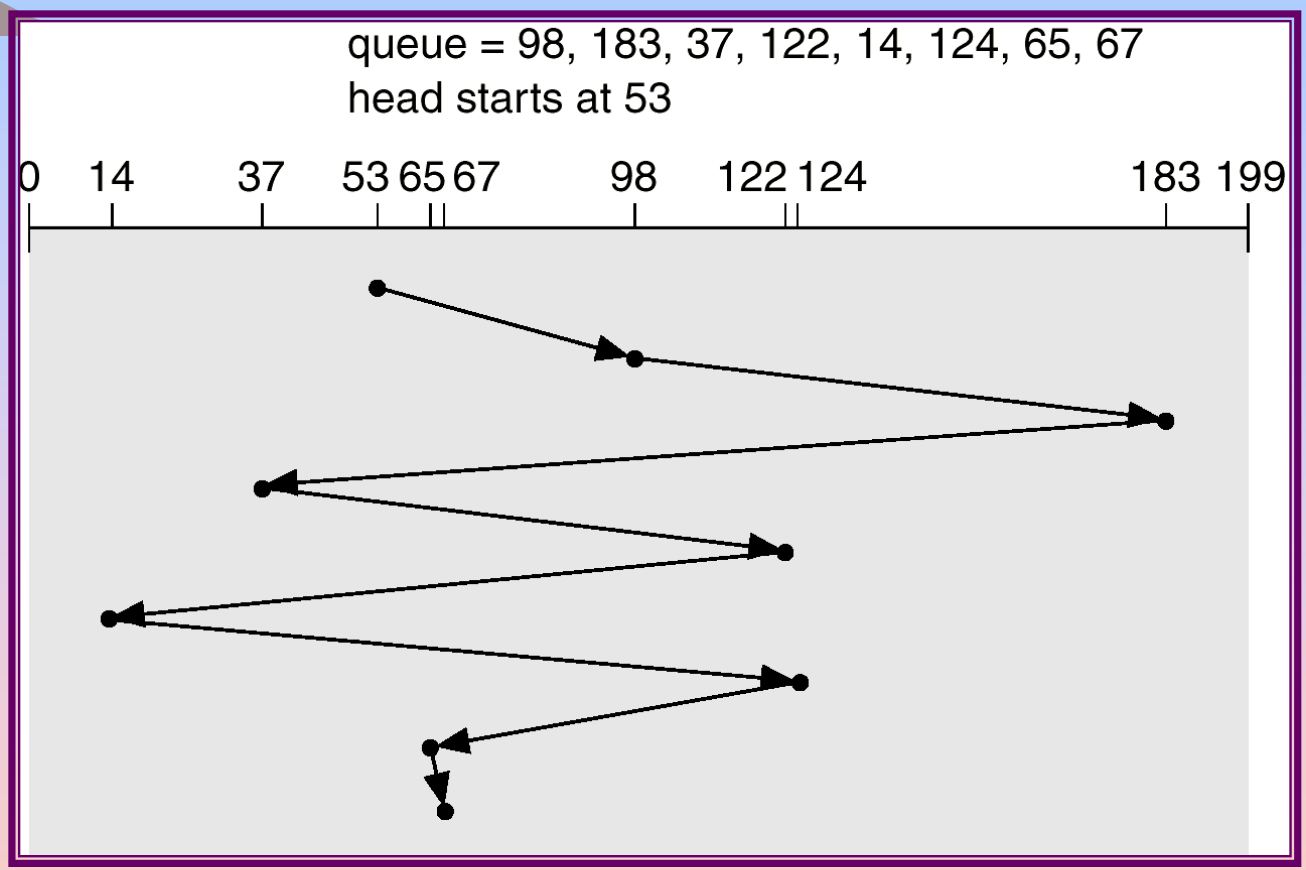
- Several algorithms exist to schedule the servicing of disk I/O requests.
- We illustrate them with a request queue (0-199).

98, 183, 37, 122, 14, 124, 65, 67

Head pointer 53



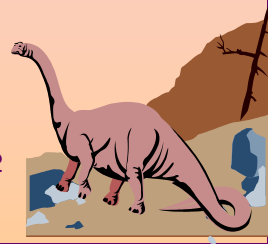
FCFS



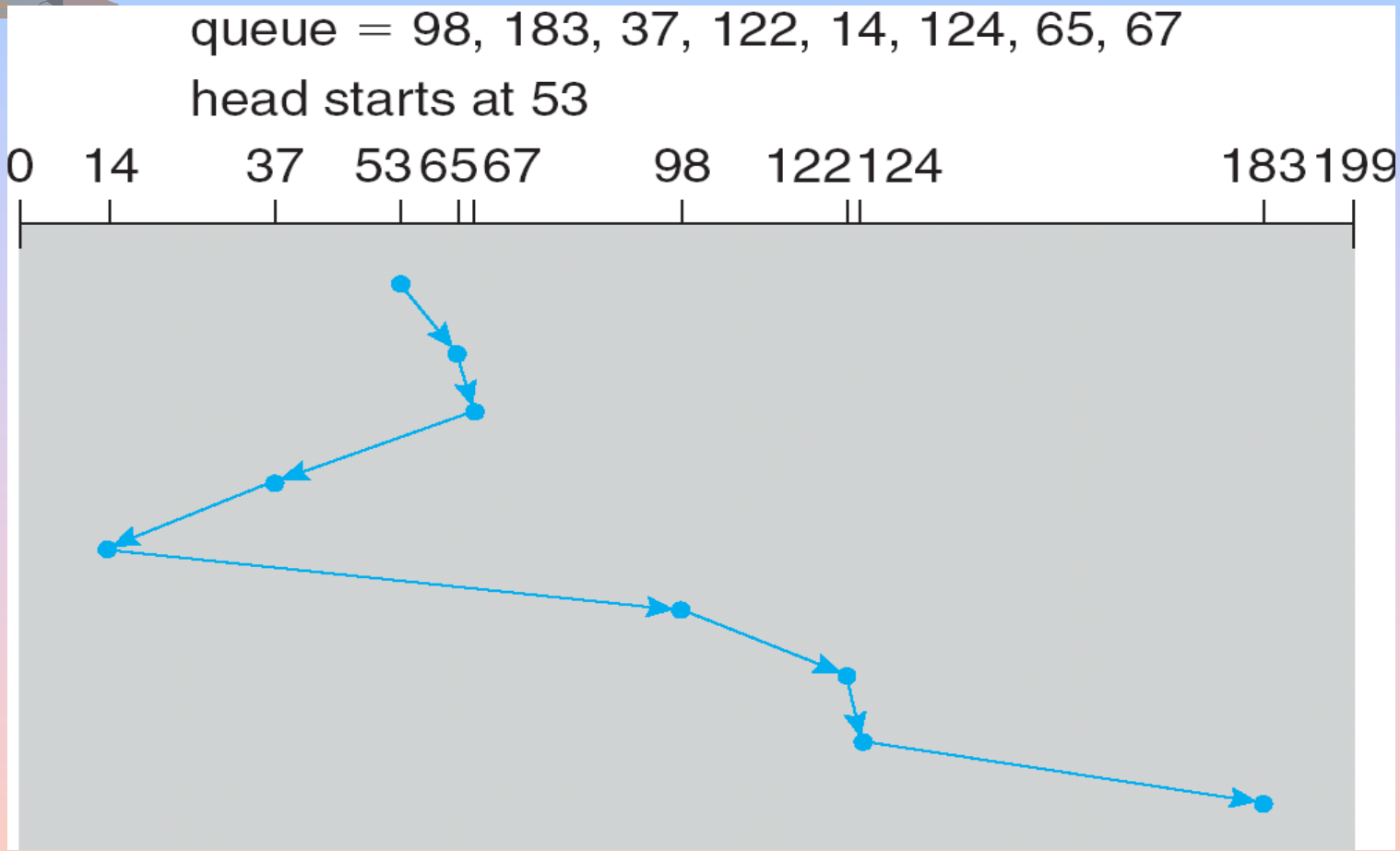
- Serving sequence: 53,98,183,37,122,14,124,65,67
- Illustration shows total head movement of 640 cylinders.

SSTF

- Selects the request with the minimum seek time from the current head position
- SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests



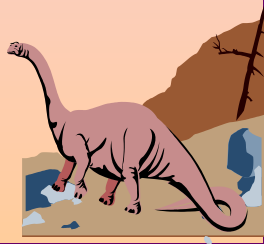
SSTF (Cont)



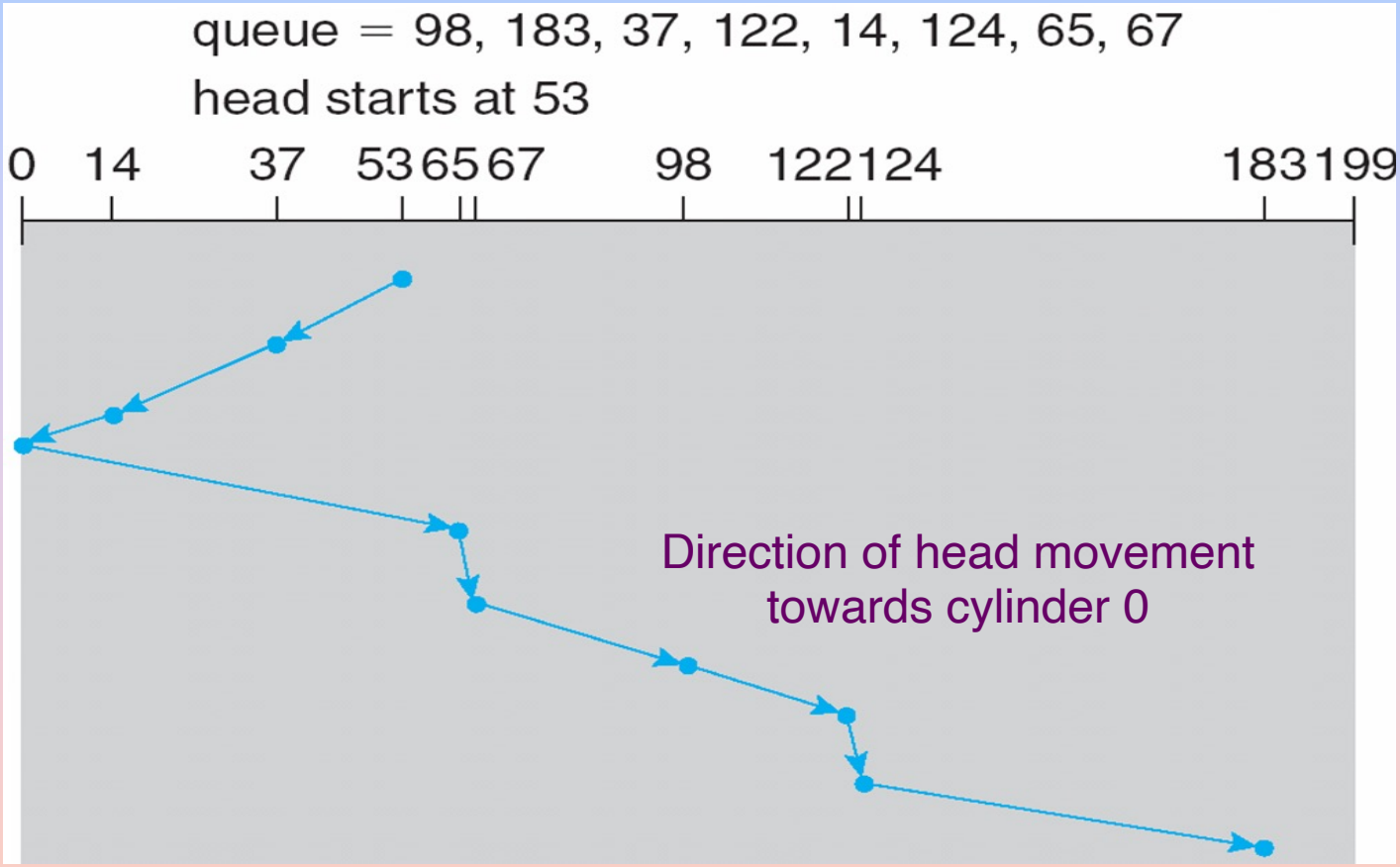
- Serving sequence: 53,65,67,37,14,98,122,124,183
- Total head movement of 236 cylinders

SCAN

- The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed, and servicing continues.
- Sometimes called the *elevator algorithm*.



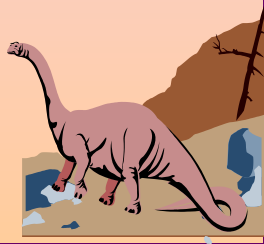
SCAN (Cont.)



- Serving sequence: 53,37,14,0,65,67,98,122,124,183
- Illustration shows total head movement of 236 cylinders.

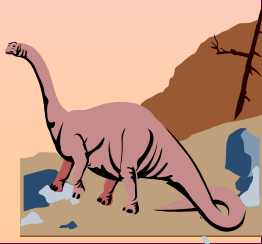
LOOK

- Version of SCAN
- Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk
- Serving Sequence:53,37,14,65,67,98,122,124,183
- total head movement of 236-28=208 cylinders

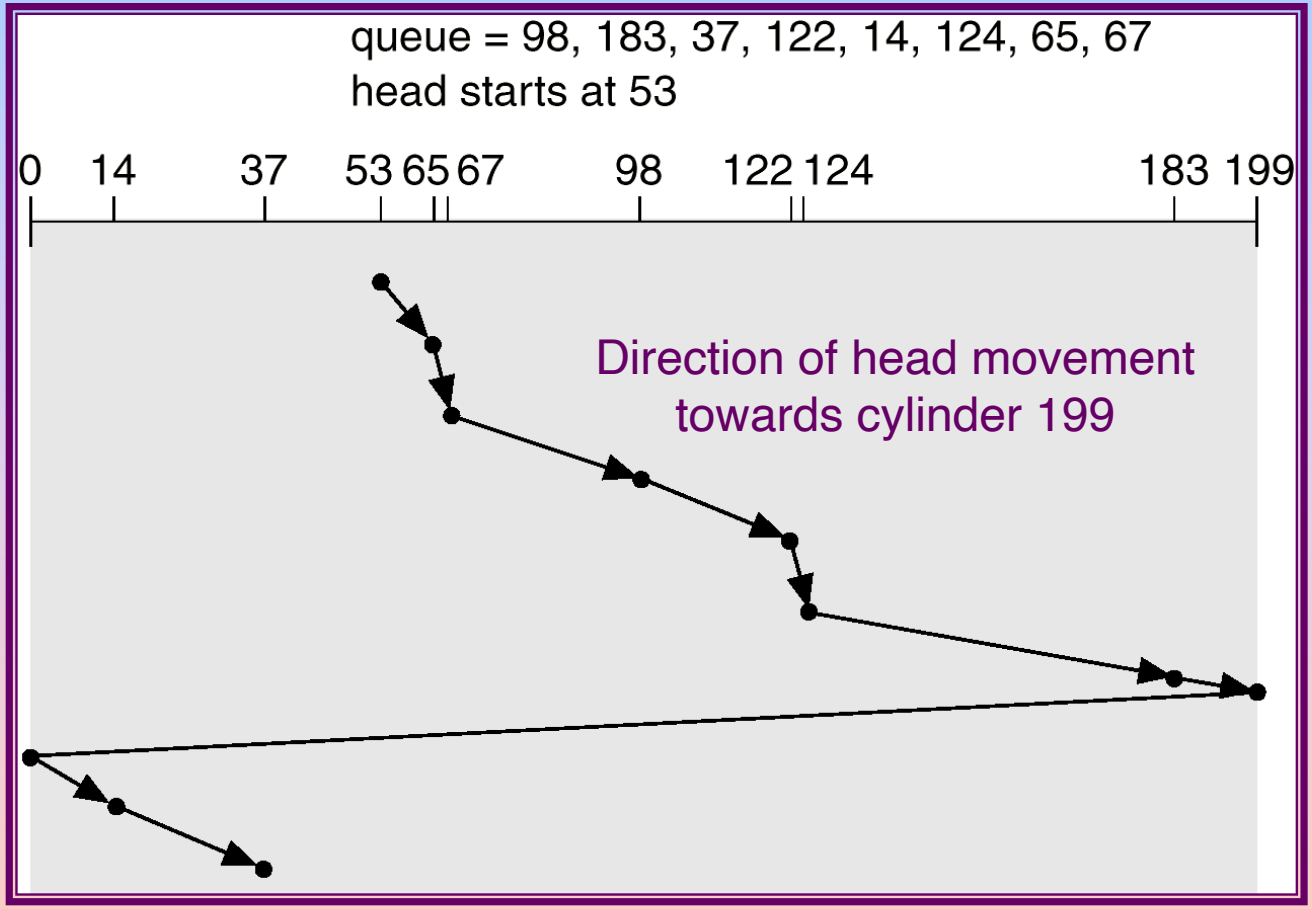


C-SCAN

- Provides a more uniform wait time than SCAN.
- The head moves from one end of the disk to the other. servicing requests as it goes. When it reaches the other end, however, it immediately returns to the beginning of the disk, without servicing any requests on the return trip.
- Treats the cylinders as a circular list that wraps around from the last cylinder to the first one.
- Thus, requests are **always** serviced in only one (the same) direction .



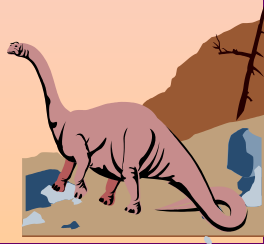
C-SCAN (Cont.)



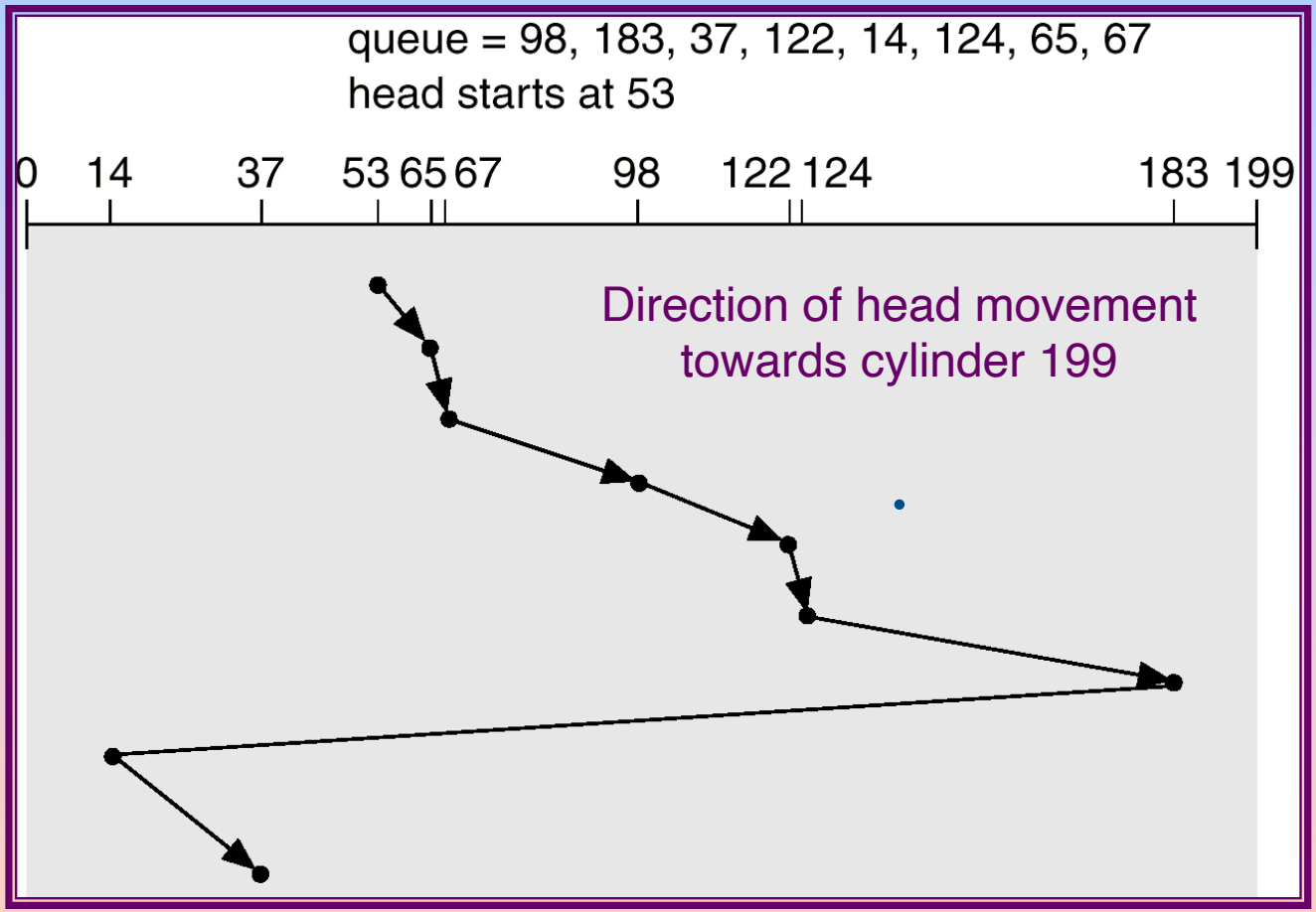
- Serving Sequence:53,65,67,98,122,124,183,199,0,14,37
- Total head movement of 183 cylinders (return trip movement=0)

C-LOOK

- Version of C-SCAN
- Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk.



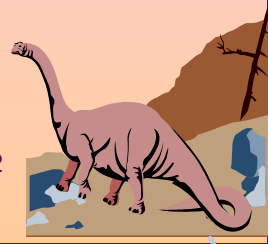
C-LOOK (Cont.)



- Serving Sequence:53,65,67,98,122,124,183,14,37
- Total head movement of 153 cylinders

Selecting a Disk-Scheduling Algorithm

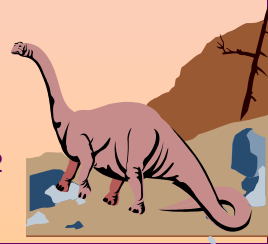
- SSTF is common and has a natural appeal
- SCAN and C-SCAN perform better for systems that place a heavy load on the disk.
- Performance depends on the number and types of requests.
- Requests for disk service can be influenced by the file-allocation method.
- The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary.
- Either SSTF or LOOK is a reasonable choice for the default algorithm.



Additional Example

Suppose that a disk drive has 200 cylinders numbered 0 to 199. The drive is currently serving a request at cylinder 100, and the previous request was at cylinder 95. The queue of pending requests in FIFO order is 55, 58,39,18,90,160,150,38,184. Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests for each of the following scheduling algorithms:

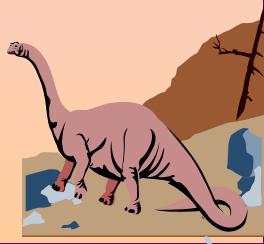
i) FCFS ii) SSTF iii) SCAN iv) LOOK v) C-SCAN vi) C-LOOK



Answer

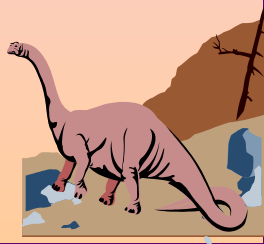
- Total Head Movement

i) FCFS : 498 Cylinders
ii) SSTF : 248 Cylinders
iii) SCAN : 280 Cylinders
iv) LOOK : 250 Cylinders
v) C-SCAN : 189 Cylinder
vi) C-LOOK : 156 Cylinders



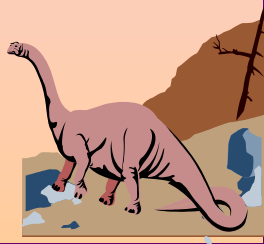
Disk Sector Queuing & Interleaving

- Disk access time=seek time + latency time
- **Disk scheduling algorithms** minimizes seek time ; given the set of requests containing cylinder numbers.
- **Queuing** and **interleaving** are methods to **minimize disk latency**



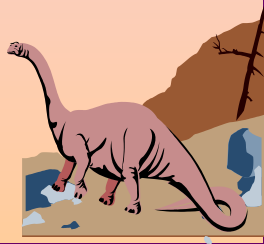
Queuing

- **Queuing** seeks to minimize latency time
- If the hardware is clever enough to know which sector is passing under the head
- reorder queue of sectors to exploit this fact (minimize latency)
- For example, if there are requests for sectors 4, 9 and 12 and the head is passing over sector 5 we can schedule 9 and 12 first



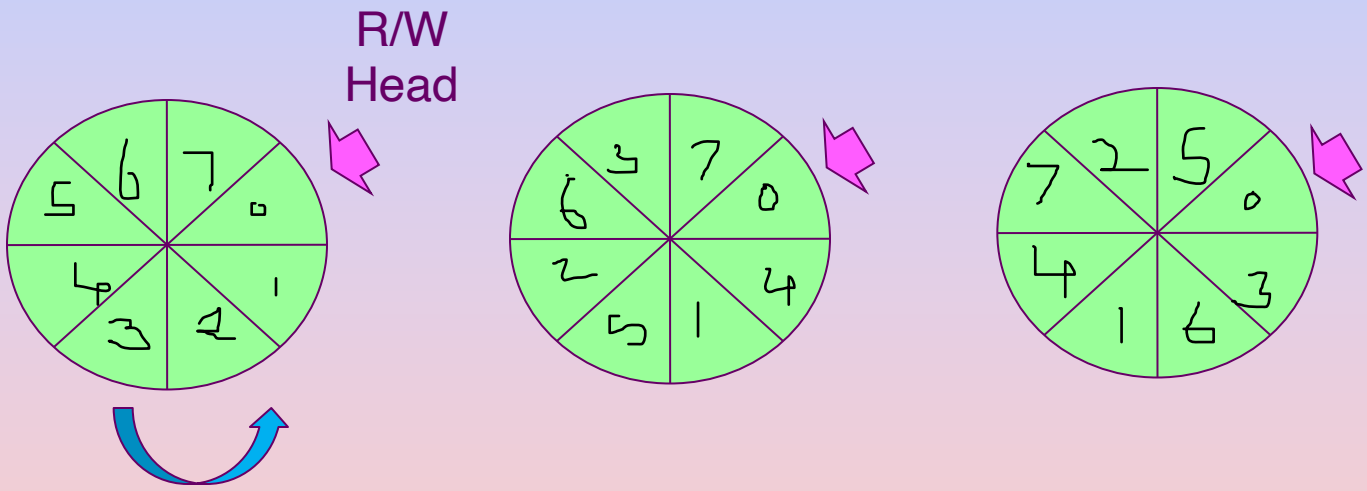
Interleaving

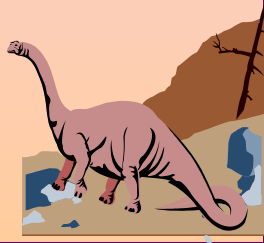
- Often the OS is too slow to process blocks that are read from consecutive locations on the disk.
- **Interleaving** seeks to minimize "unnecessary" latency resulting from additional platter revolutions when the OS is unprepared to receive more data on consecutive disk locations
- It is the process of reordering the sectors so as to allow the OS time for processing



Interleaving(Contd..)

Non-Interleaved Single Interleaved Double Interleaved





THANK YOU