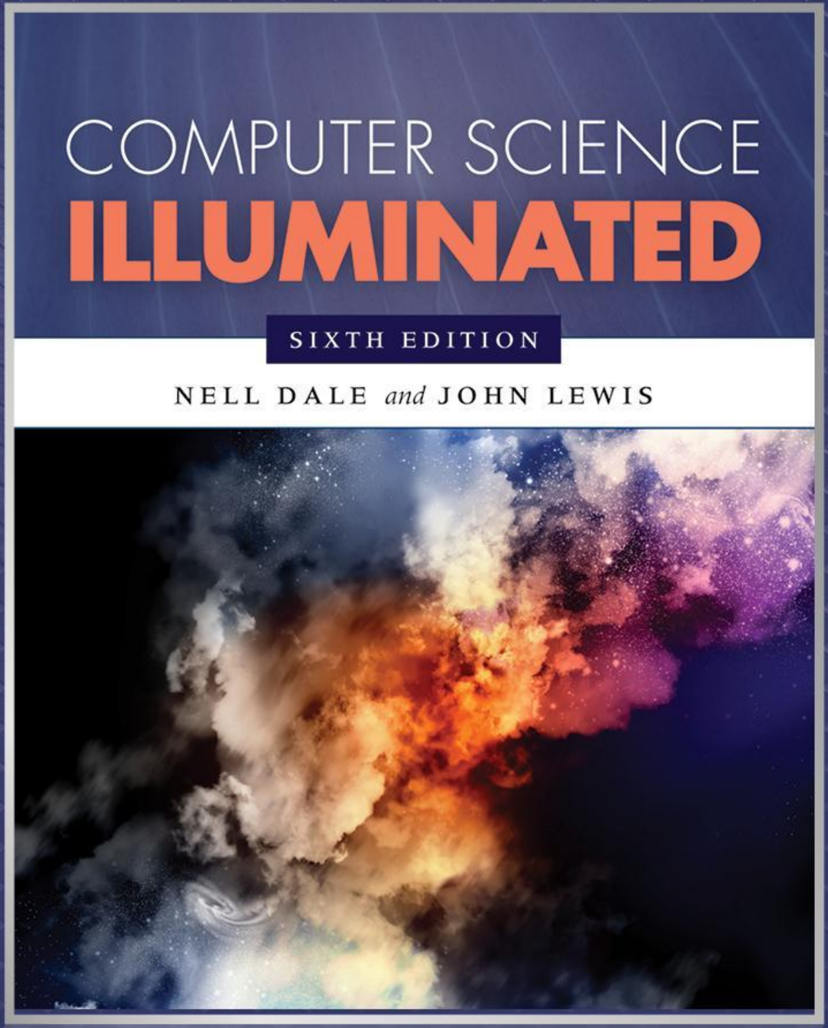
File Systems and Directories



Edits by Michael Edwards and Chris Whyley

Chapter Goals

- Describe the purpose of files, file systems, and directories
- Distinguish between text and binary files
- Identify various file types by their extensions
- Explain how file types improve file usage
- Define the basic operations on a file

Chapter Goals

- Compare and contrast sequential and direct file access
- Discuss the issues related to file protection
- Describe a directory tree
- Create absolute and relative paths for a directory tree
- Describe several disk-scheduling algorithms

File Systems

File

A named collection of related data, used for organizing secondary memory

File system

The operating system's logical view of the files it manages

Directory

A named group of files

Text and Binary Files

Text file

A file that contains characters from the ASCII or Unicode character sets

Binary file

A file that contains data in a specific format, requiring special interpretation of its bits

Text and Binary Files

The terms text file and binary file are somewhat misleading

Ultimately, all information on a computer is stored as binary digits

Text files are formatted as chunks of 8 or 16 bits, interpreted as characters

Binary files are formatted in some other special format

File Types

File Type

The **kind** of information contained in a document

Most files, regardless of format, contain a specific type of information

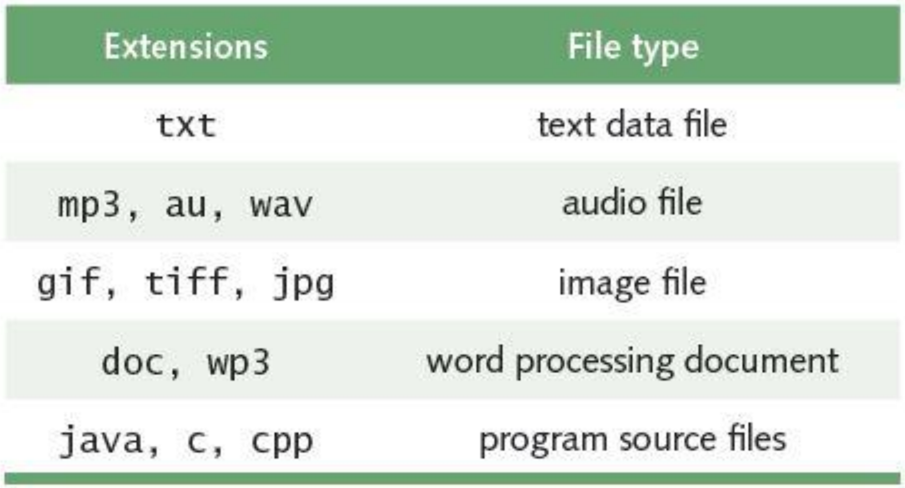
File extension

Part of a file name that indicates the type

File names are often in two parts:

File name . File Extension

File Types



Extensions	File type
txt	text data file
mp3, au, wav	audio file
gif, tiff, jpg	image file
doc, wp3	word processing document
java, c, cpp	program source files

FIGURE 11.1 Some common file types and their extensions

What kinds of files are the following?

- *Chapter.doc*
- *Figure1.jpg*
- *Interview.wav*
- *MyFavorite.mp3*

What's the advantage of using the appropriate extension?

File Operations

What operations do you think you might want to perform on or with a file?

File Access

Sequential access

The technique in which data in a file is accessed in a linear fashion; to get the last record, you must read all of the records

Direct access

The technique in which data in a file is conceptually divided into numbered logical records and accessed directly, by specifying logical record numbers

File Access

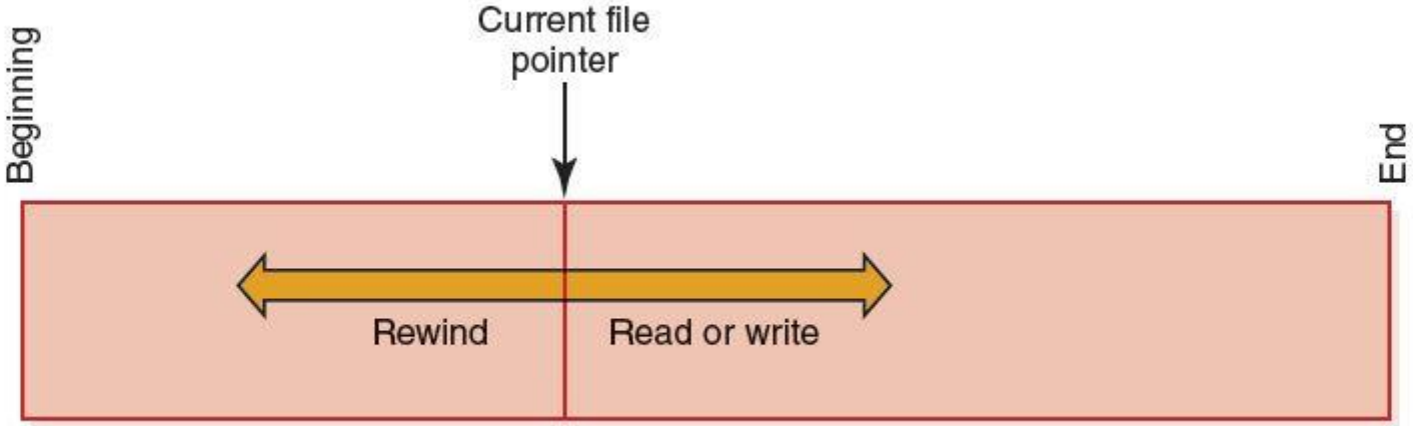


FIGURE 11.2 Sequential file access

File Access

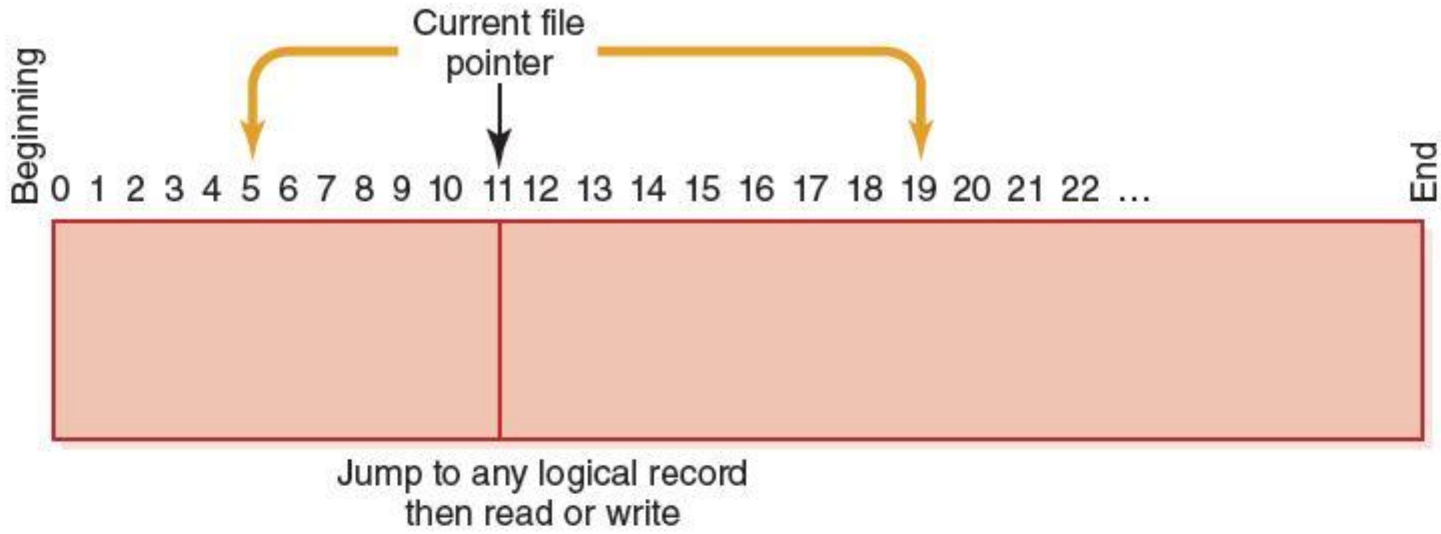


FIGURE 11.3 Direct file access

File Access

Which file access do you think is easier to implement: sequential access or direct access?

File Protection

File protection

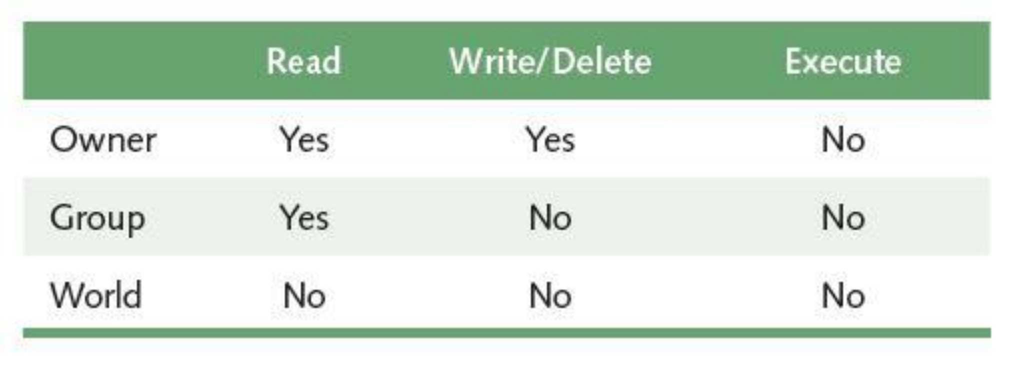
The process of limiting file access

- In multiuser systems, file protection is of primary importance
- We don't want one user to be able to access another user's files unless the access is specifically allowed
- A file protection mechanism determines who can use a file and for what general purpose

Why is file protection important?
Give two examples

File Protection

An example of a file protection scheme is the file settings in the Unix operating system, which are divided into three categories

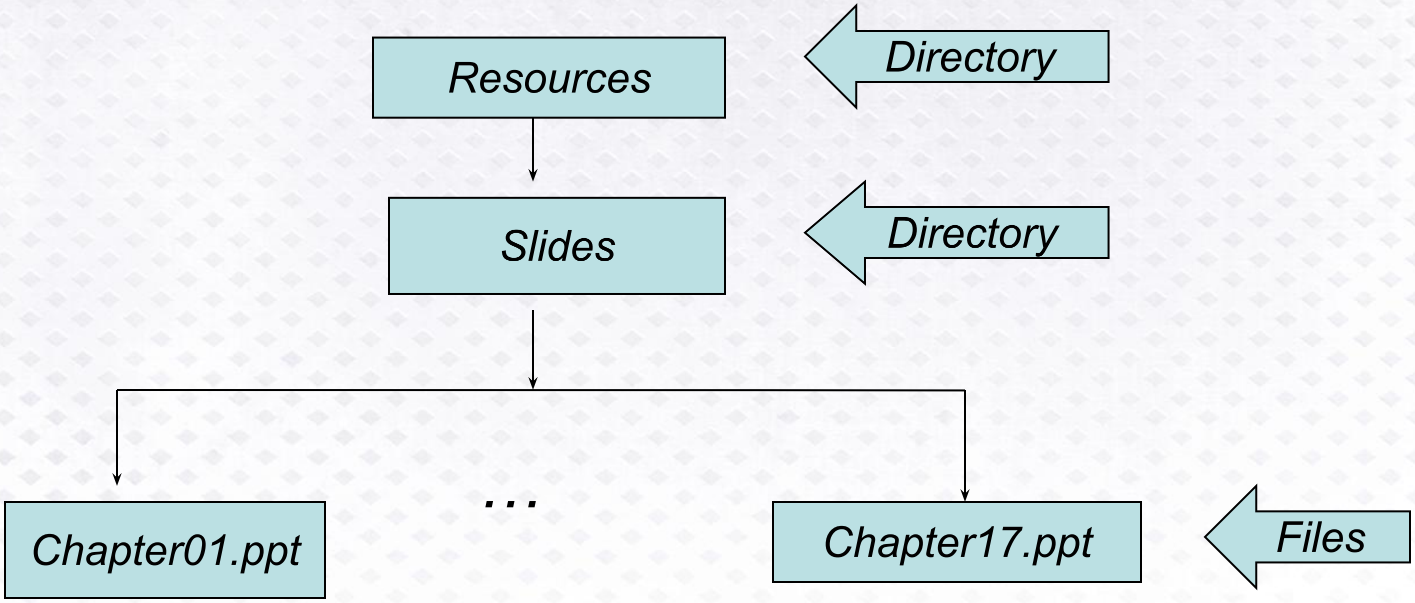


	Read	Write/Delete	Execute
Owner	Yes	Yes	No
Group	Yes	No	No
World	No	No	No

Directory Trees

Recall that a directory is a named group of files.

A directory can be contained within another directory



Directory Trees

Parent directory

The containing directory

Subdirectory

The directory being contained

Directory tree

A logical view of a file system; a structure showing the nested directory organization of a file system

Root directory

The directory at the highest level

Directory Trees

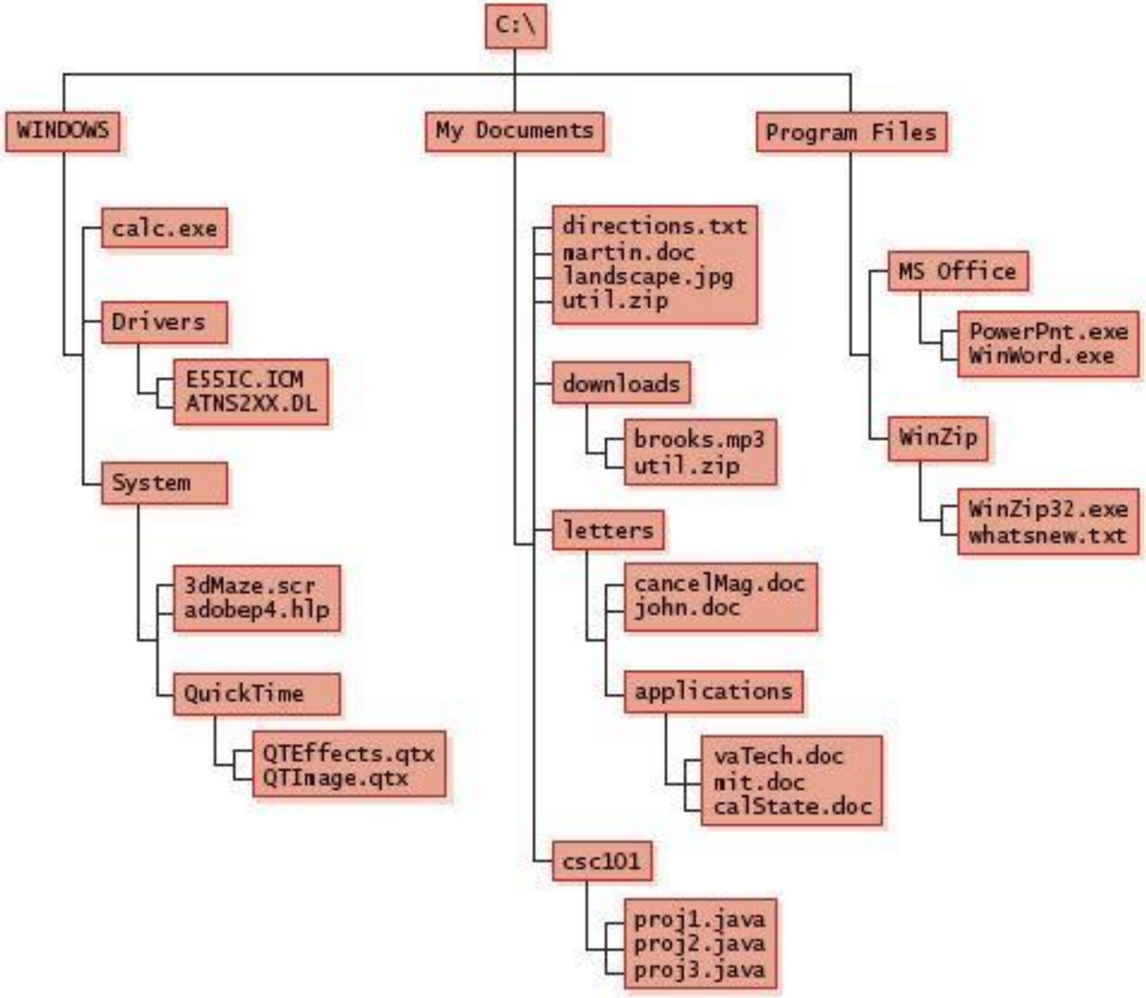


FIGURE 11.4. A Windows directory tree

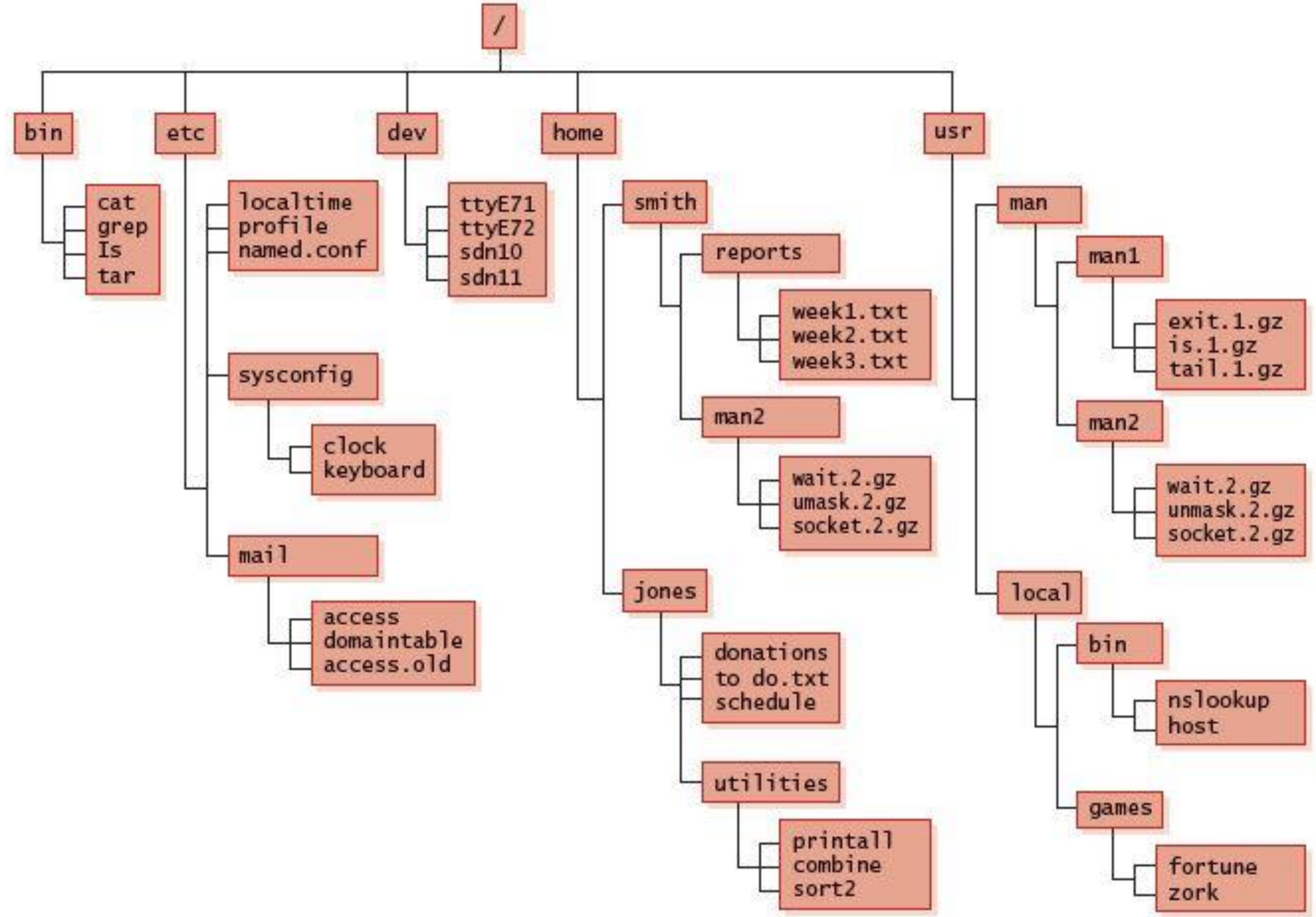
Directory Trees

At any point in time, you can be thought of as working in a particular location (that is, a particular subdirectory)

Working directory

The subdirectory in which you are working

A Unix Directory Tree



- /
  - bin
    - cat
    - grep
    - Is
    - tar
  - etc
    - localtime
    - profile
    - named.conf
    - sysconfig
      - clock
      - keyboard
    - mail
      - access
      - domaintable
      - access.old
  - dev
    - ttyE71
    - ttyE72
    - sdn10
    - sdn11
  - home
    - smith
      - reports
        - week1.txt
        - week2.txt
        - week3.txt
      - man2
        - wait.2.gz
        - umask.2.gz
        - socket.2.gz
    - jones
      - donations
      - to do.txt
      - schedule
      - utilities
        - printall
        - combine
        - sort2
  - usr
    - man
      - man1
        - exit.1.gz
        - is.1.gz
        - tail.1.gz
      - man2
        - wait.2.gz
        - unmask.2.gz
        - socket.2.gz
    - local
      - bin
        - nslookup
        - host
      - games
        - fortune
        - zork

FIGURE 11.5 A UNIX directory tree

Path Names

Path

A text designation of the location of a file or subdirectory in a file system

Absolute path

A path that begins at the root and includes all successive subdirectories

Relative path

A path name that begins at the current working directory

Path Names

Absolute paths

C:\Program Files\MS Office\WinWord.exe

C:\My Documents\letters\applications\vaTech.doc

C:\Windows\System\QuickTime

If current working directory is

C:\My Documents\letters

Relative paths

cancelMag.doc

applications\calState.doc

Distinguish between absolute and relative paths

File Systems

File Systems are influenced by the physical nature of disks (remember back to CS-150)

A large disk drive may actually be a disk pack with a number of platters

File Systems

Each physical disk (or platter) is divided into concentric rings (tracks) with each track divided into sectors

Reading and writing is usually done by sector (not individual bytes)

File Systems

It is important to note

- The majority of files are small
- A few files are large

In order to handle both efficiently, an OS may choose to use a larger block size than the physical disk sector size

- May increase transfer efficiency
- Might be convenient to match page size

File Systems

Three common approaches:

- Contiguous
- Chained
- Indexed

Contiguous

Allocate as a collection of adjacent blocks

Maintain a list of unallocated blocks

Advantages:

Easy access (sequential and random)

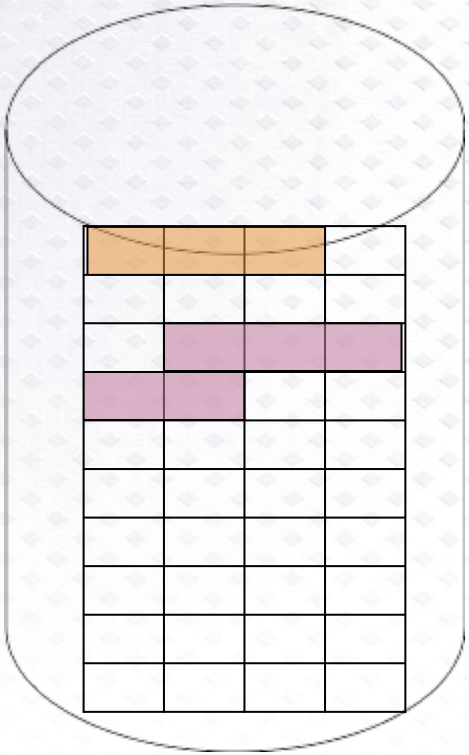
Simple, few seeks

Disadvantages:

External fragmentation

May not know file size in advance

Name	Start	Length
File1	0	3
File2	9	5



Chained

Mark allocated blocks as used

Link with pointers

<u>Advantages:</u>

No external fragmentation

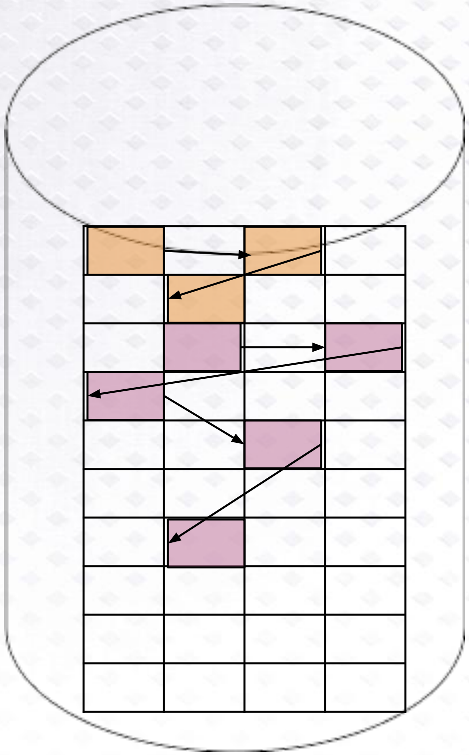
Files can grow easily

<u>Disadvantages:</u>

Lots of seeking

Random access is difficult

Name	Start	Length
File1	0	3
File2	9	5



FAT File Systems

- The File Allocation Table (FAT) contains an entry for every block on a disk.
- Directory entry for a file contains address of first block in file
- The FAT entry for this block points to next block

FAT File Systems

- Blank entries in FAT indicate free blocks (no need for free list)

- Tendency to use larger block sizes
  - NTFS uses 4K block size for disks larger than 2GB
  - FAT32 uses 4K up to 8GB, 8K up to 16GB, 16K up to 32GB, and 32K above 32GB

- The larger the block size the smaller the FAT

Indexed

Allocate array of pointers during creation

Advantages:

Small internal fragmentation

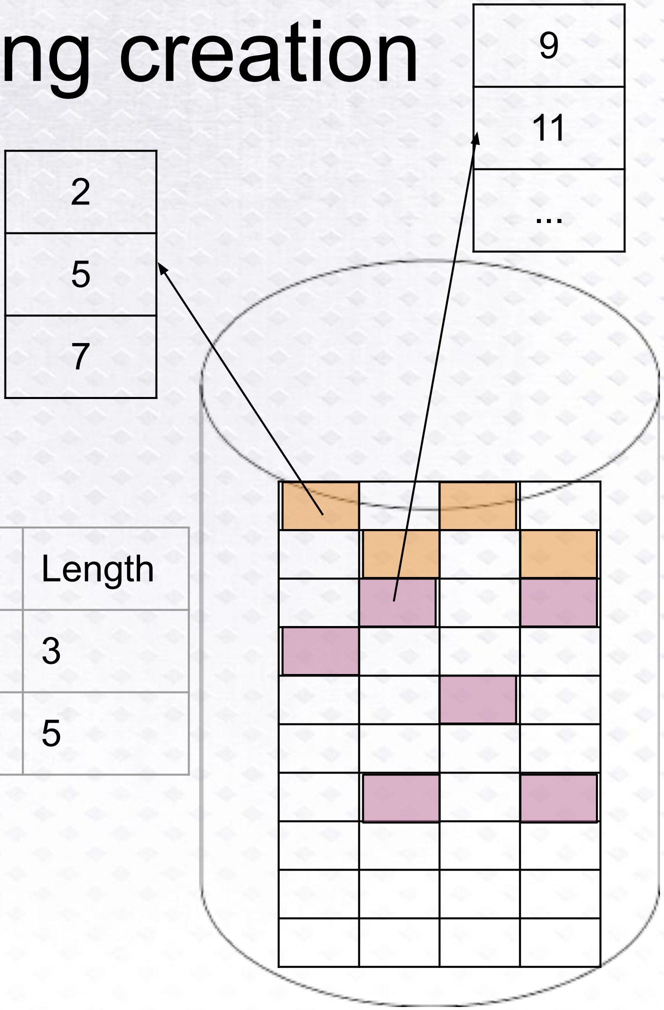
Easy sequential and random access

Disadvantages:

Lots of seeks

Hard size limit (unless an index block is able to link to another

Name	Start	Length
File1	0	3
File2	9	5



Disk Scheduling

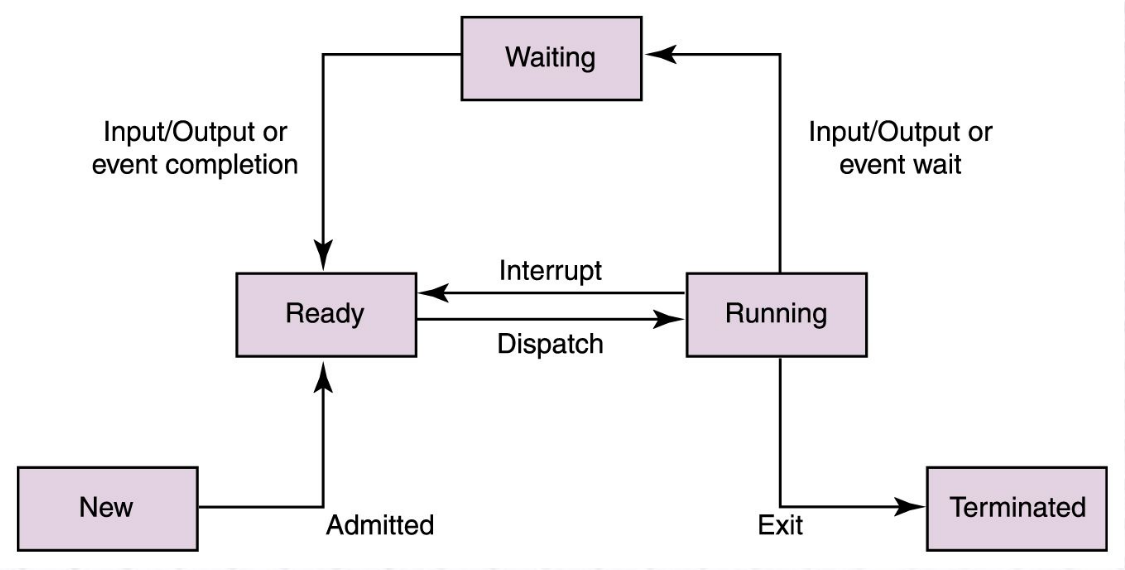
As a computer deals with multiple processes over a period of time, a list of requests to access the disk builds up

Disk scheduling

The technique that the operating system uses to determine which requests to satisfy first

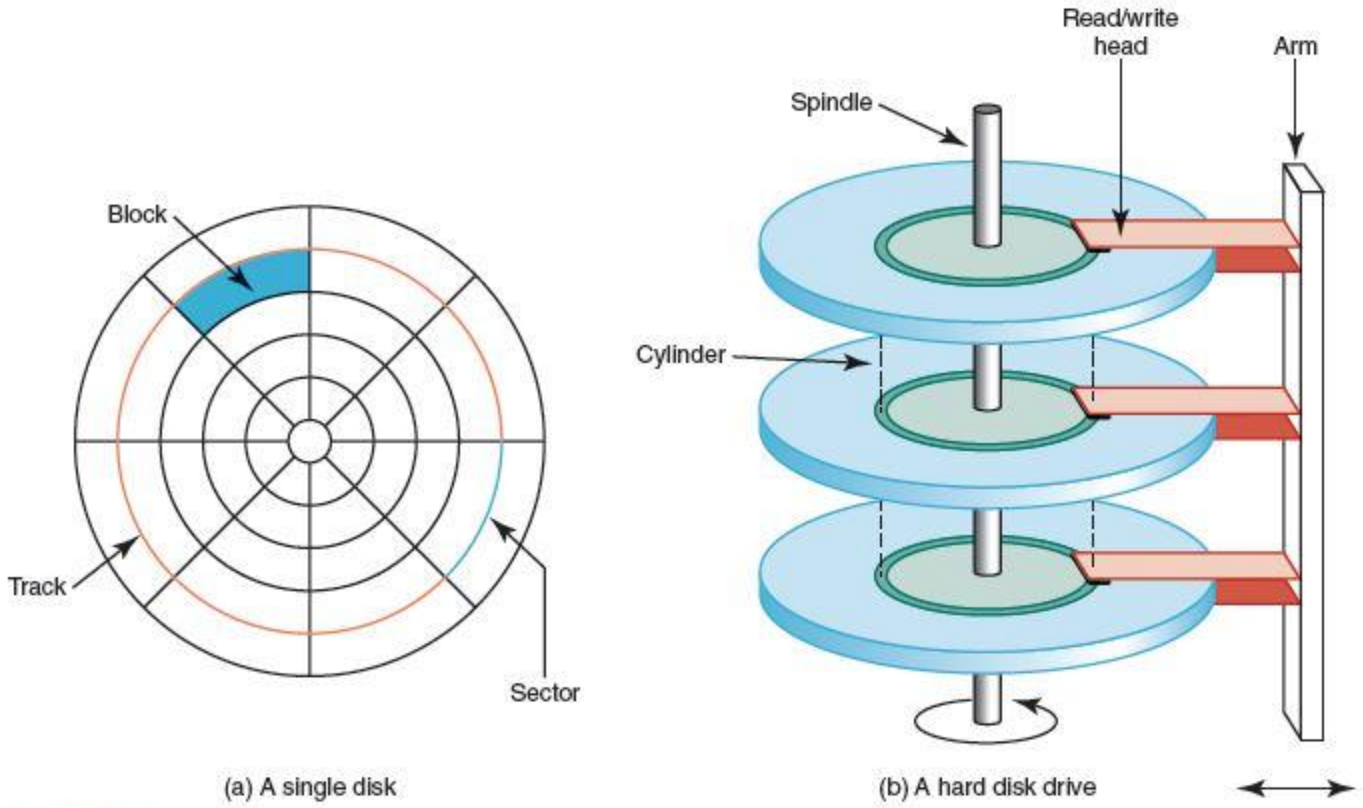
Disk Scheduling

Which of these requests should be performed next?



Recall that I/O is the slowest aspect of any computing system

Disk Scheduling



(a) A single disk

(b) A hard disk drive

FIGURE 11.6 A magnetic disk drive.

Remember seek time and latency?

Disk Scheduling

First-Come, First-Served (FCFS)

Sound familiar?

Requests are serviced in the order they arrive, without regard to the current position of the heads

Shortest-seek-time-first (SSTF)

Disk heads are moved the minimum amount possible to satisfy a pending request

Scan

Disk heads continuously move in and out servicing requests as they are encountered

Disk Scheduling Examples

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

Seek time = time to get read/write head over correct track (we will presume 0.1ms per track)

Latency = time for sector to be in correct position (we will presume 1ms)

Disk Scheduling: FCFS

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26

Seek Time:

Disk Scheduling: FCFS

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 49

Seek Time: 2.3

Disk Scheduling: FCFS

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 49 91

Seek Time: 2.3 + 4.2

Disk Scheduling: FCFS

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 49 91 22

Seek Time: 2.3 + 4.2

Disk Scheduling: FCFS

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 49 91 22 61

Seek Time: 2.3 + 4.2 + 6.9

Disk Scheduling: FCFS

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 49 91 22 61 7

Seek Time: 2.3 + 4.2 + 6.9 + 3.9

Disk Scheduling: FCFS

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 49 91 22 61 7 62

Seek Time: 2.3 + 4.2 + 6.9 + 3.9 + 5.4

Disk Scheduling: FCFS

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 49 91 22 61 7 62 33

Seek Time: 2.3 + 4.2 + 6.9 + 3.9 + 5.4 + 5.5

Disk Scheduling: FCFS

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 49 91 22 61 7 62 33 35

Seek Time: 2.3 + 4.2 + 6.9 + 3.9 + 5.4 + 5.5 + 2.9

Disk Scheduling: FCFS

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 49 91 22 61 7 62 33 35

Seek Time: 2.3 + 4.2 + 6.9 + 3.9 + 5.4 + 5.5 + 2.9 + 0.2

Disk Scheduling: FCFS

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 49 91 22 61 7 62 33 35

Seek Time: 2.3 + 4.2 + 6.9 + 3.9 + 5.4 + 5.5 + 2.9 + 0.2
= 31.3 ms

Latency: 8 x 1ms = 8 ms
Total: 8 + 31.3 ms = 39.3ms

Disk Scheduling: SSTF

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26

Seek Time:

Disk Scheduling: SSTF

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 22

Seek Time: 0.4

Disk Scheduling: SSTF

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 22 33

Seek Time: 0.4 + 1.1

Disk Scheduling: SSTF

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 22 33 35

Seek Time: 0.4 + 1.1 + 0.2

Disk Scheduling: SSTF

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 22 33 35 49

Seek Time: 0.4 + 1.1 + 0.2 + 1.4

Disk Scheduling: SSTF

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 22 33 35 49 61

Seek Time: 0.4 + 1.1 + 0.2 + 1.4 + 1.2

Disk Scheduling: SSTF

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 22 33 35 49 61 62

Seek Time: 0.4 + 1.1 + 0.2 + 1.4 + 1.2 + 0.1

Disk Scheduling: SSTF

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 22 33 35 49 61 62 91

Seek Time: 0.4 + 1.1 + 0.2 + 1.4 + 1.2 + 0.1 + 2.9

Disk Scheduling: SSTF

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 22 33 35 49 61 62 91 7

Seek Time: 0.4 + 1.1 + 0.2 + 1.4 + 1.2 + 0.1 + 2.9 + 8.4

Disk Scheduling: SSTF

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 22 33 35 49 61 62 91 7

Seek Time: 0.4 + 1.1 + 0.2 + 1.4 + 1.2 + 0.1 + 2.9 + 8.4
= 15.7 ms

Latency: 8 x 1ms = 8 ms
Total: 8 + 15.7 ms = 23.7ms

Disk Scheduling: SSTF

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26

26 22 33 35 49 61 62 91 7

Seek Time: 0.4 + 1.1 + 0.2 + 1.4 + 1.2 + 0.1 + 2.9 + 8.4
= 15.7 ms

Latency: 8 x 1ms = 8 ms
Total: 8 + 15.7 ms = 23.7ms

But requests for extreme/edge tracks could be delayed indefinitely!

Disk Scheduling

SCAN Disk Scheduling works like an elevator

- An elevator is designed to visit floors that have people waiting. In general, an elevator moves from one extreme to the other (say, the top of the building to the bottom), servicing requests as appropriate
- The SCAN disk-scheduling algorithm works in a similar way, except instead of moving up and down, the read/write heads move in toward the spindle, then out toward the platter edge, then back toward the spindle, and so forth

Disk Scheduling: SCAN

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26 moving towards 1

26

Seek Time:

Disk Scheduling: SCAN

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26 moving towards 1

26 22

Seek Time: 0.4

Disk Scheduling: SCAN

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26 moving towards 1

26 22 7

Seek Time: 0.4 + 1.5

Disk Scheduling: SCAN

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26 moving towards 1

26 22 7 33

Seek Time: 0.4 + 1.5 + 2.6

Disk Scheduling: SCAN

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26 moving towards 1

26 22 7 33 35

Seek Time: 0.4 + 1.5 + 2.6 + 0.2

Disk Scheduling: SCAN

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26 moving towards 1

26 22 7 33 35 49

Seek Time: 0.4 + 1.5 + 2.6 + 0.2 + 1.4

Disk Scheduling: SCAN

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26 moving towards 1

26 22 7 33 35 49 61

Seek Time: 0.4 + 1.5 + 2.6 + 0.2 + 1.4 + 1.2

Disk Scheduling: SCAN

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26 moving towards 1

26 22 7 33 35 49 61 62

Seek Time: 0.4 + 1.5 + 2.6 + 0.2 + 1.4 + 1.2 + 0.1

Disk Scheduling: SCAN

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26 moving towards 1

26 22 7 33 35 49 61 62 91

Seek Time: 0.4 + 1.5 + 2.6 + 0.2 + 1.4 + 1.2 + 0.1 + 2.9

Disk Scheduling: SCAN

Ordered cylinder requests: 49, 91, 22, 61, 7, 62, 33, 35

Read/write heads at Cylinder 26 moving towards 1

26 22 7 33 35 49 61 62 91

Seek Time: 0.4 + 1.5 + 2.6 + 0.2 + 1.4 + 1.2 + 0.1 + 2.9
= 10.3 ms

Latency: 8 x 1ms = 8 ms
Total: 8 + 10.3 ms = 18.3ms

Ethical Issues

Spam

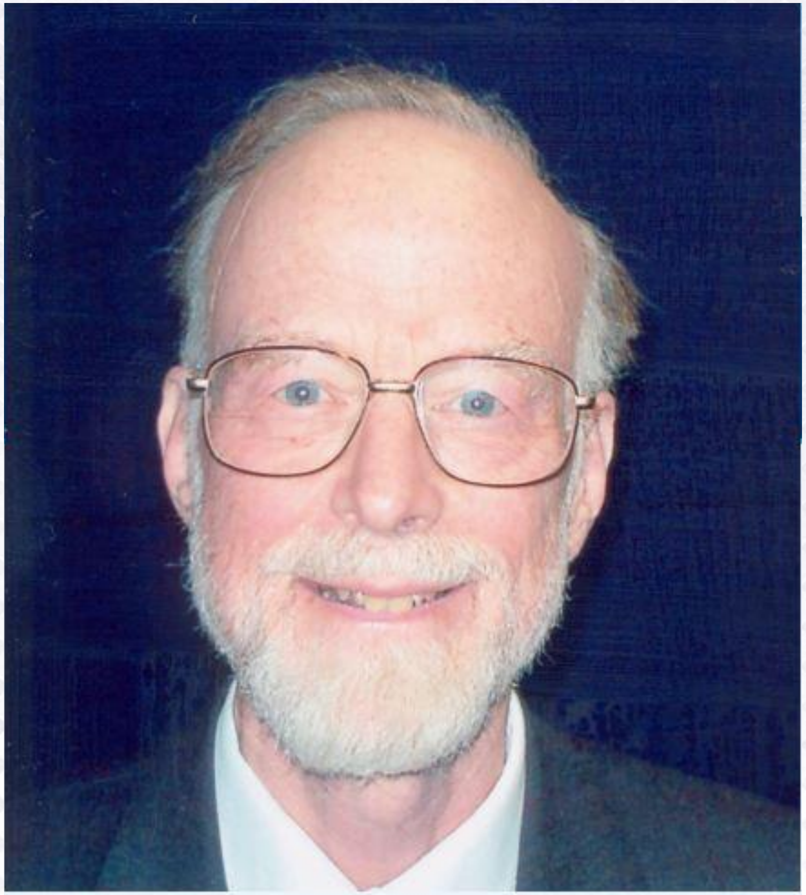
How many spam messages have you received today?

Why is spam expensive?

Have you ever sent spam?

Can you explain the CAN-SPAM Act?

Who am I?



Courtesy of Inamori Foundation

I was awarded Knighthood in 1999.
What university did I retire from and where am I working now?

Do you know?

What is an RFID tag? What is it used for?

Why did people voluntarily let a club owner implant them with RFIDs?

What is HRV? How can software be used to monitor and reduce stress?

What is the eNeighbor System?

How can it be used to monitor conditions of the elderly?