Administrivia

- Reminder: Homework 5 due Wednesday.

Slide 1

Minute Essay From Last Lecture

- One person asked about ongoing research in memory management. Good question.

- The discussion in the textbook (section 3.8) confirms what would have been my guess — at least of the time of its writing, memory management was pretty a solved problem, though there's some work being done in adapting to changes in hardware, such as SSDs.

Slide 2

Files and Filesystems — Overview

- Very abstract view — requirements for long-term information storage are:

    - Store large amounts of information.

    - Have information survive past end of creating process.

    - Allow concurrent access by multiple processes.

- Usual solution — "files" on disk and other external media, organized into "file systems".

- In terms of the two views of an O/S:

    - "Virtual machine" view — filesystem is important abstraction.

    - "Resource manager" view — filesystem manages disk (and other I/O device) resources.

- We'll look first at the user view, then at implementation.

Slide 3

File Abstraction

- Many, many aspects of "file abstraction" — name, type, ownership, etc., etc. Most involve choices/tradeoffs.

- In the following slides, a quick tour of some of the major ones, with some of the possible variations.

Slide 4

File Abstraction, Continued

- File names — always "text string", but some choices: maximum length? case-sensitive? ASCII or Unicode? "extension" required?

- File structure — how file appears to application program:

  - Unstructured sequence of bytes — maximum flexibility, but maybe more work for application.

  Slide 5

  - Sequence of fixed-length records — widely used in older systems, not much any more.

  - Tree (or other) structure supporting access by key.

File Abstraction, Continued

- File types — include "regular files", also directories and (in some systems, such as UNIX) "special files". Regular files subdivide into:

  - ASCII files — sequences of ASCII characters, generally separated into lines by line-end character(s).

  Slide 6

  - Binary files — everything else, including executables, various archives, MS Word format, etc., etc. Most have some structure, defined by the expectations of the program(s) that work with them — applications for some types, operating system for executables.

- File access — sequential versus random-access.

- File attributes — "other stuff" associated with file (owner, protection info, time of creation / last use, etc.)

Slide 7

File Abstraction, Continued

- File operations (things one can do to a file) include create, delete, open, close, read, write, get attributes, set attributes. Example program using low-level wrappers for system calls on p. 274.

- Many systems also support operations for "memory-mapped files" (read whole file into memory, process there, write back out — as mentioned in previous discussion of memory management).

Slide 8

Directory/Folder Abstraction

- Basic idea — way of grouping / keeping track of files. Can be
  - Single-level (simple but restrictive).
  - Two-level (almost as simple, better than single-level if multiple users, but also restrictive).
  - Hierarchical.

- Implies need for path names, which can be absolute or relative (to "working directory").

- "Hierarchical" implies a tree structure, but one could include support for something to allow a more-general directed graph (more later). Might be useful as a way to easily share files among users.

- Operations on directories include create, delete, open, close, read, add entry, remove entry, link, unlink.

Slide 9

Minute Essay

- Anything noteworthy about Homework 4 (turned in last week)?