Administrivia

- Reminder: Homework 3 due Wednesday.

- Remaining quizzes . . . — I propose that we have five not six (I would still drop the lowest one) and that we schedule them for the next two Mondays.

Slide 1

Minute Essay From Last Lecture

- The point of the story from long ago: A system that's spending too much time paging isn't going to be doing much else.

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.

Slide 3

- 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.

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

Slide 5

- 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.
  - 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

Slide 6

- 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).
  - 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.)

File Abstraction, Continued

Slide 7

- 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).

Directory/Folder Abstraction

Slide 8

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

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

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

Slide 9

Minute Essay

- None — quiz.