IOWA STATE UNIVERSITY

Department of Electrical and Computer Engineering

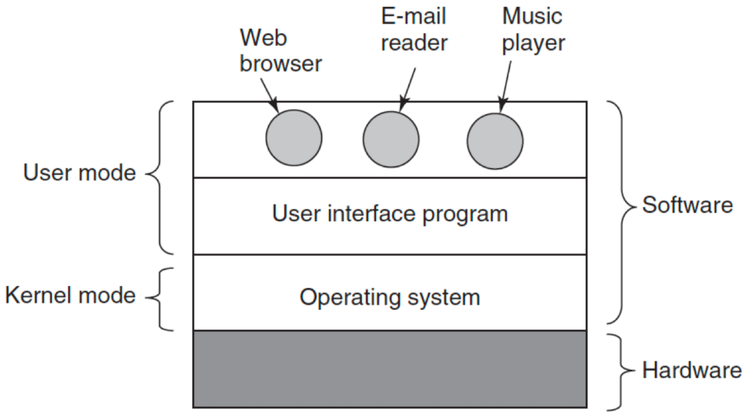
Lecture 03:
OS Introduction II

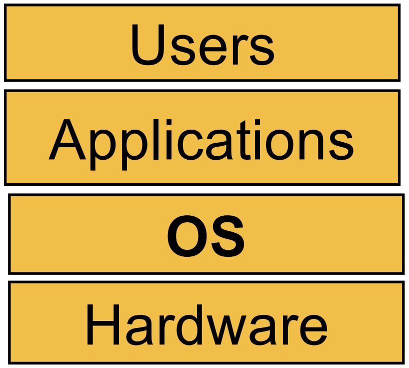
Agenda

- Recap
- OS Introduction II
  - Computer Hardware Review II
  - OS Abstractions for HW

Recap

- Computer System Structure
  - OS: b/w Hardware & Apps
    - manage hardware
    - provide services to apps/users

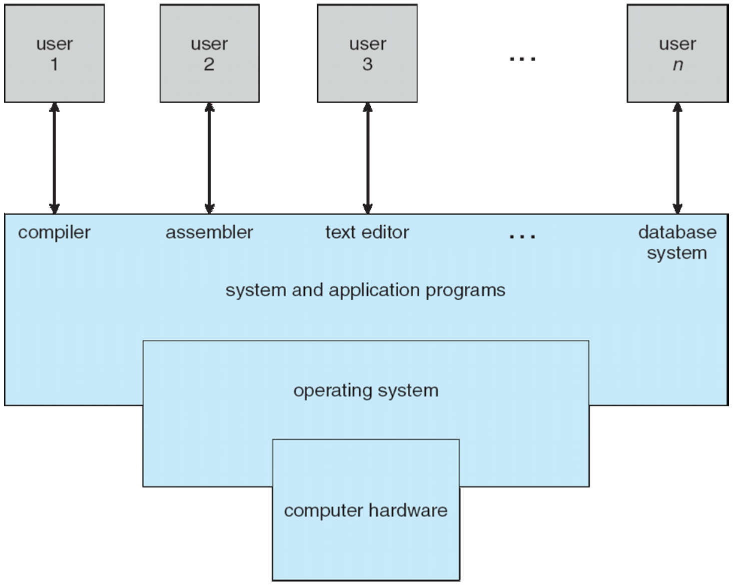



Users

Applications

OS

Hardware

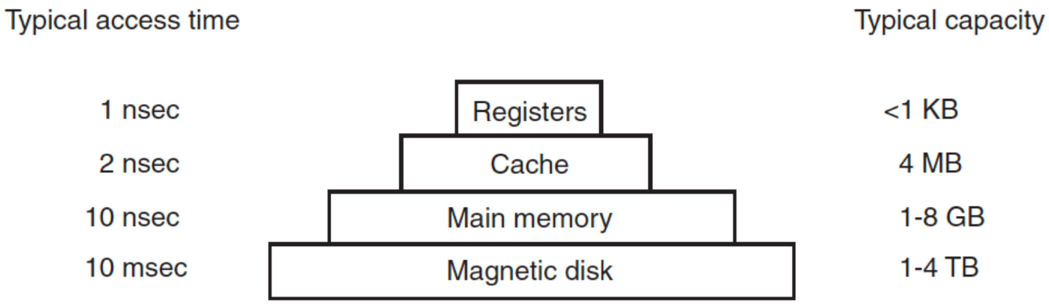


Recap

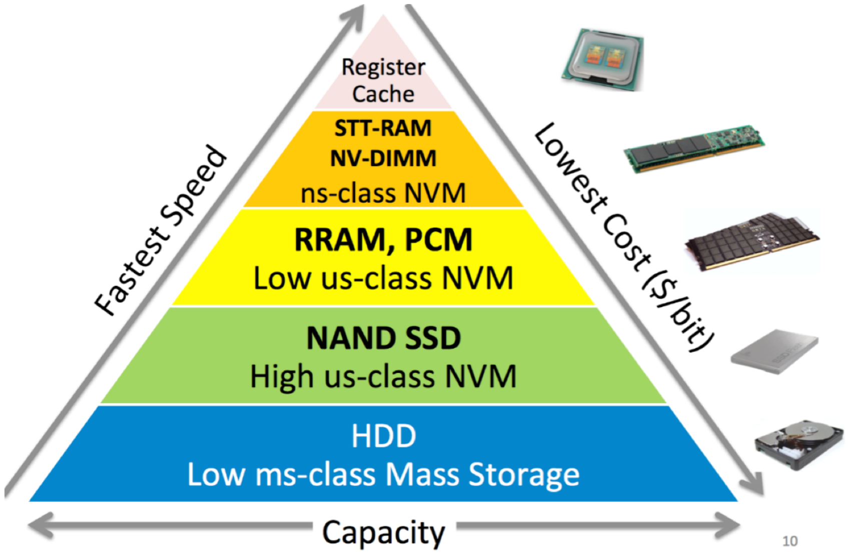
- Computer Hardware Review I
  - CPU
    - executes a set of instructions
    - some important registers
      - **program counter**
      - **stack pointer**
      - **PSW** (Program Status Word)
      - **performance counters**
    - common features
      - pipeline
      - superscalar
      - multi-core
      - cache

Recap

- Computer Hardware Review I
  - Memory
    - hierarchy with tradeoffs
      - latency, capacity, persistency, cost, …
      - Non-volatile memories are revolutionizing computers



Typical access time		Typical capacity
1 nsec	Registers	<1 KB
2 nsec	Cache	4 MB
10 nsec	Main memory	1-8 GB
10 msec	Magnetic disk	1-4 TB



Register
Cache

STT-RAM
NV-DIMM
ns-class NVM

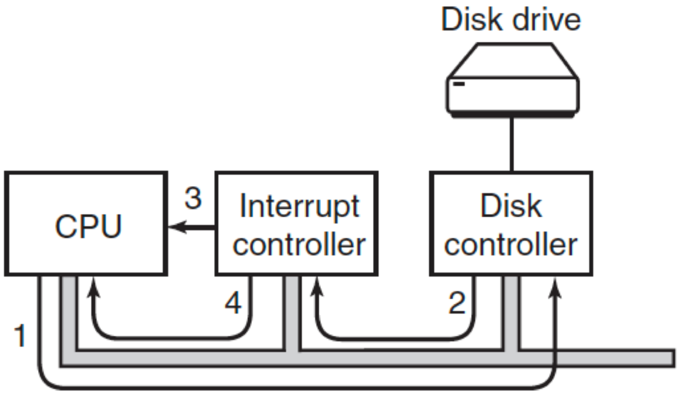
RRAM, PCM
Low us-class NVM

NAND SSD
High us-class NVM

HDD
Low ms-class Mass Storage

Recap

- Computer Hardware Review I
    - I/O devices
        - three ways of communication
            - Busy waiting
            - Interrupt
            - DMA

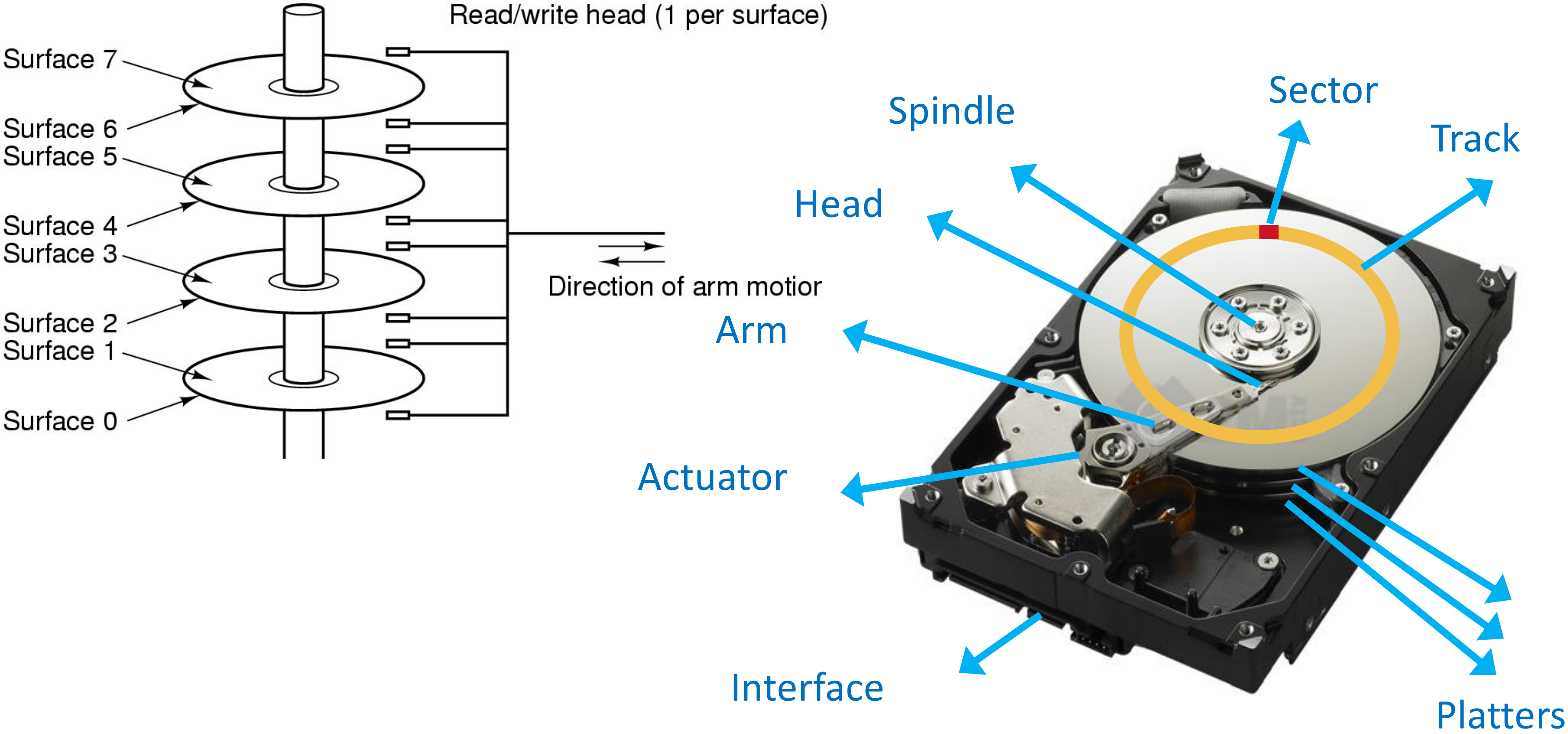


Agenda

- ~~Recap~~
- OS Introduction II
  - Computer Hardware Review II
  - OS Abstractions for HW

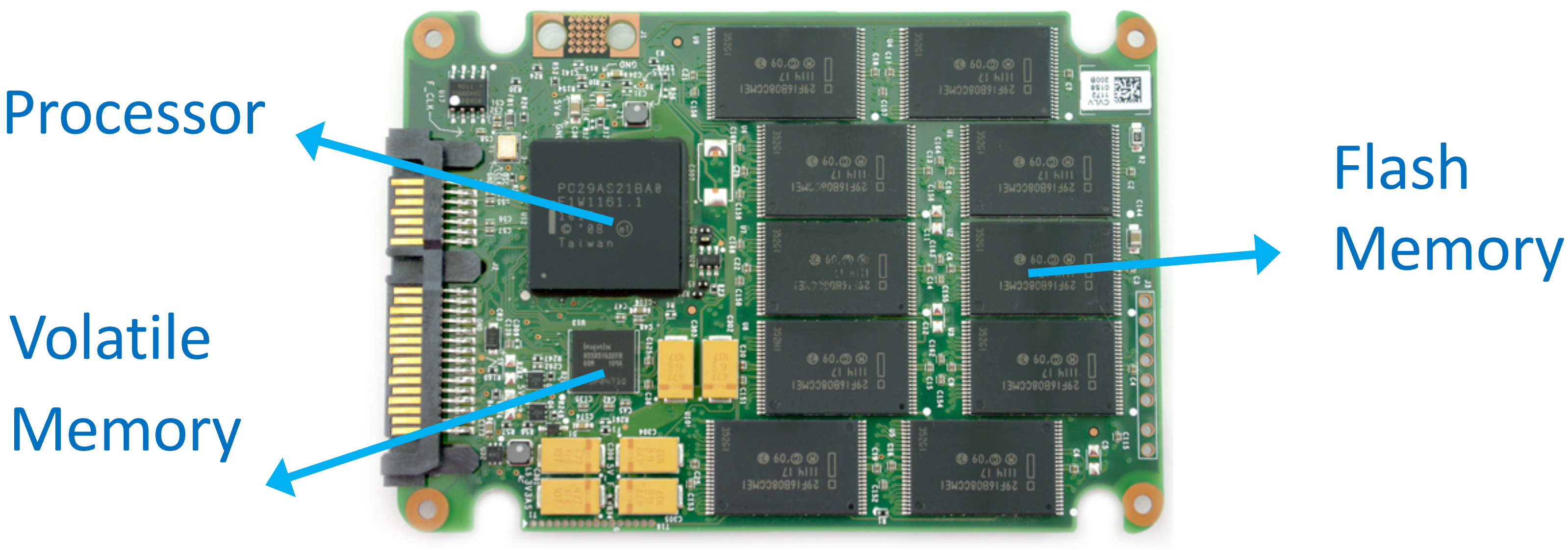
Computer Hardware Review II

- I/O devices
  - Example: hard disk drives (HDD)



Computer Hardware Review II

- I/O devices
  - Example: solid state drives (SSD)
    - Intel 710 Enterprise SSD



Processor

Flash Memory

Volatile Memory

    - ‘A disruptive technology’

Agenda

- ~~Recap~~
- OS Introduction II
  - ~~Computer Hardware Review II~~
  - OS Abstractions for HW

OS Abstractions

- Common OS Abstractions for HW
  - CPU
    - process and/or thread
  - Memory
    - address space
  - Disks
    - files

OS Abstractions

- Common OS Abstractions for HW
  - CPU
    - process and/or thread
  - Memory
    - address space
  - Disks
    - files
- Advantages?

OS Abstractions

- Common OS Abstractions for HW
  - CPU
    - process and/or thread
  - Memory
    - address space
  - Disks
    - files
- Advantages?
  - Allow applications to reuse common facilities
  - Make different devices look the same
  - Provide higher-level or more useful functionality

OS Abstractions

- Common OS Abstractions for HW
  - CPU
    - process and/or thread
  - Memory
    - address space
  - Disks
    - files
- Challenges?

OS Abstractions

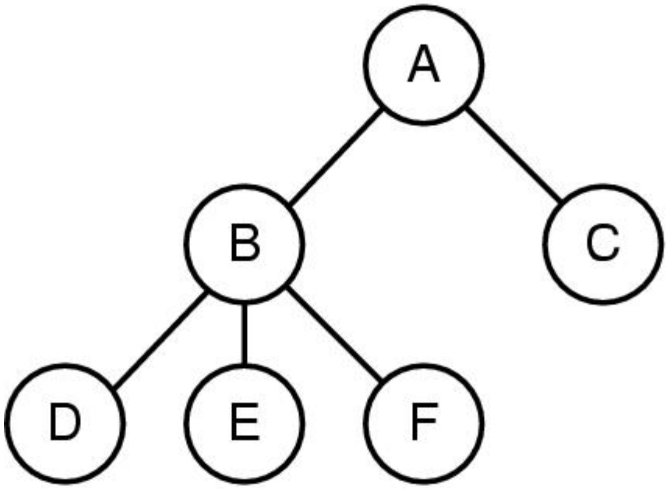
- Common OS Abstractions for HW
  - CPU
    - process and/or thread
  - Memory
    - address space
  - Disks
    - files
- Challenges?
  - What are the correct abstractions?
  - How much of hardware should be exposed?
  - Tradeoffs among different goals?

Process

- Process: A program *in execution*
  - passive (program) V.S. active (process)
  - a unit of work within the system
  - needs resources to accomplish task
    - CPU, memory, I/O device, etc.
  - Fundamentally, a container that holds all the information needed to run a program.
    - Address space
      - Program (text), data, stack
    - Register values
      - Program counter, stack pointer, etc

Process

- Process Tree
  - In UNIX-like OSes, one process may “spawn” more processes
    - E.g., Process A created two child processes, B and C; Process B created three child processes, D, E, and F



Process

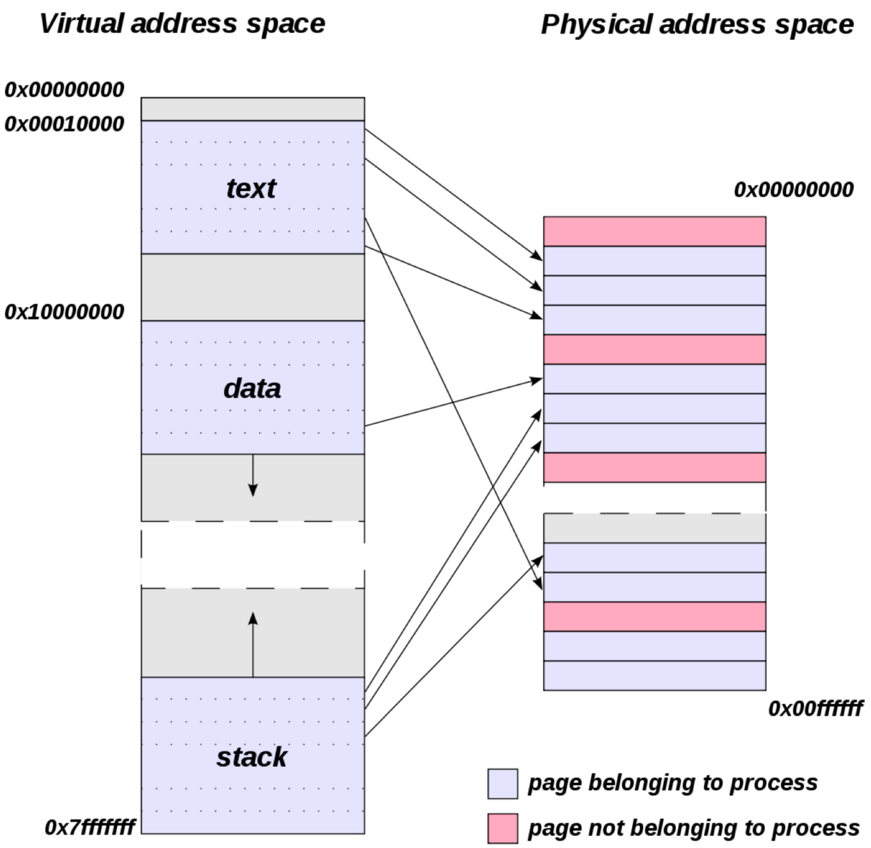
- Process management
  - Typically, a system has many processes running concurrently on CPUs
    - Multiplexing CPUs
  - A scheduler decides which process to run next among all the current processes
  - OS kernel is responsible for:
    - Creating/killing processes
    - Suspending/resuming processes
    - Providing mechanisms for
      - process communication, synchronization, ...

Address Space

- The physical memory is *an array of bytes*
  - a *shared* resource, managed by the OS
- A program keeps all of its data structures in memory
  - **Read memory** (load):
    - Specify an address to be able to access the data
  - **Write memory** (store):
    - Specify the data to be written to the given address

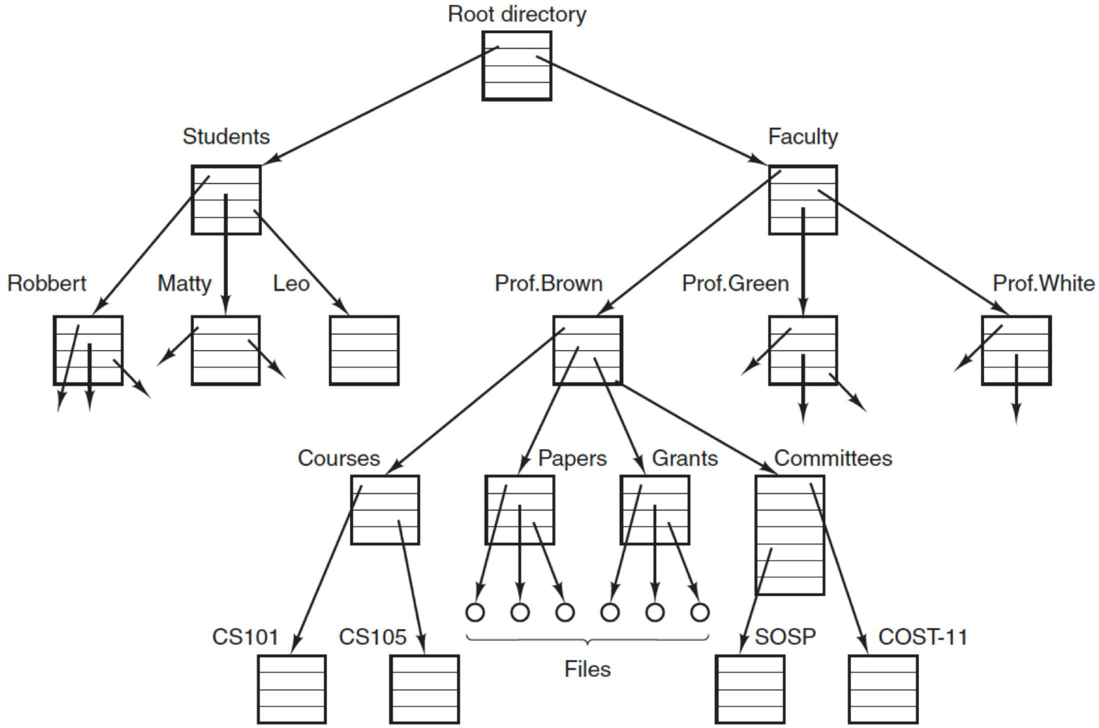
Address Space

- Each process accesses its own private **virtual address space**.
  - OS maps address space onto the physical memory
  - A memory reference within one running program does not affect the address space of other processes



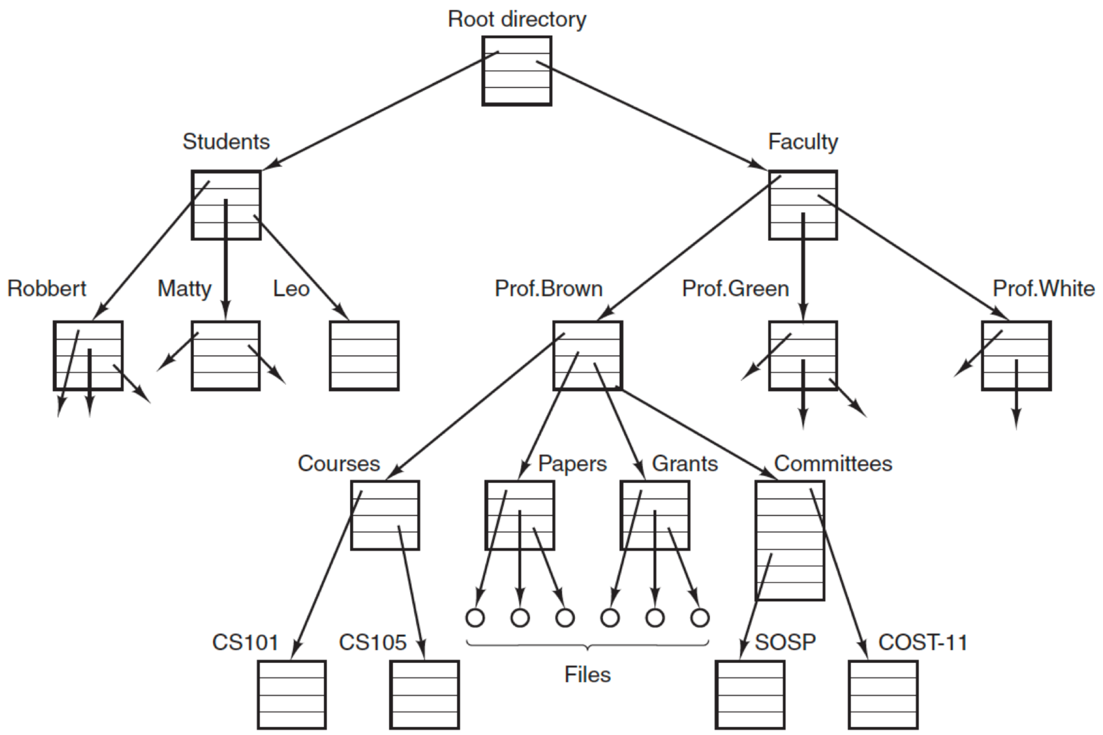
Files

- “Everything is a file” on Linux
- A logical view of a file system tree
  - Directory contains (sub)directories and files



Files

- Every file can be specified by giving its path from the root
  - E.g., /Faculty/Prof.Brown/Courses/CS101



Files

- Special files are provided to represent I/O devices
  - block special files for block devices
    - E.g., disks
  - character special files for character devices
    - E.g., printer
- File systems are responsible for managing the files
  - E.g., Ext4, NTFS

Agenda

- ~~Recap~~
- ~~OS Introduction II~~
  - ~~Computer Hardware Review II~~
  - ~~OS Abstractions for HW~~

Questions?

*acknowledgement: slides include content from “Modern Operating Systems” by A. Tanenbaum, “Operating Systems Concepts” by A. Silberschatz etc., “Operating Systems: Three Easy Pieces” by R. Arpaci-Dusseau etc., and anonymous pictures from internet.