Questions answered in this lecture:
- What will you do in this course?
- What is an OS and why do you want one?
- Why study operating systems?

To do:
- Take a look at course web page and first programming project

WHAT IS AN OPERATING SYSTEM?
Not easy to define precisely…
Operating System (OS):
Software that converts hardware into a useful form for applications

WHAT DOES OS PROVIDE?
Role #1: Abstraction - Provide standard library for resources
- What is a resource?
  - Anything valuable (e.g., CPU, memory, disk)
- What abstraction does modern OS typically provide for each resource?
  - CPU: process and/or thread
  - Memory: address space
  - Disk: files
- Advantages of OS providing abstraction?
  - Allow applications to reuse common facilities
  - Make different devices look the same
  - Provide higher-level or more useful functionality
- Challenges
  - What are the correct abstractions?
  - How much of hardware should be exposed?

Role #2: Resource management – Share resources well
- Advantages of OS providing resource management?
  - Protect applications from one another
  - Provide efficient access to resources (cost, time, energy)
  - Provide fair access to resources
- Challenges
  - What are the correct mechanisms?
  - What are the correct policies?
OS ORGANIZATION

• How to cover all the topics relevant to operating systems?

THREE PIECES: FIRST

• Virtualization
  ◦ Make each application believe it has each resource to itself

  • Demo
    ◦ Virtualize CPU and memory

THREE PIECES: SECOND

• Concurrency:
  Events are occurring simultaneously and may interact with one another
• OS must be able to handle concurrent events
  ◦ Easier case
    ◦ Hide concurrency from independent processes
  ◦ Trickier case
    ◦ Manage concurrency with interacting processes
      ◦ Provide abstractions (locks, semaphores, condition variables, shared memory, critical sections) to processes
      ◦ Ensure processes do not deadlock
  ◦ Demo
    ◦ Interacting threads must coordinate access to shared data

THREE PIECES: THIRD

• Persistence: Access information permanently
  ◦ Lifetime of information is longer than lifetime of any one process
  ◦ Machine may be rebooted, machine may lose power or crash unexpectedly

• Issues:
  ◦ Provide abstraction so applications do not know how data is stored: Files, directories (folders), links
  ◦ Correctness with unexpected failures
  ◦ Performance: disks are very slow; many optimizations needed!

• Demo
  ◦ File system does work to ensure data updated correctly
ADVANCED TOPICS

- Current systems
  - Multiprocessors
  - Networked and distributed systems
  - Virtual machines

WHY STUDY OPERATING SYSTEMS?

Build, modify, or administer an operating system

Understand system performance
- Behavior of OS impacts entire machine
- Tune workload performance
- Apply knowledge across many layers
  - Computer architecture, programming languages, data structures and algorithms, and performance modeling

Fun and challenging to understand large, complex systems

TO DO

Take a look at course web page
Take a look at first programming project
Watch video of previous discussion section before Wednesday’s discussion section