CSE 120
Principles of Operating Systems

Fall 2020

Lecture 1: Course Introduction

Geoffrey M. Voelker
Lecture 1 Overview

• Class overview
• Administrative info
• Introduction to operating systems

• Go ahead and ask questions in chat
  ♦ I’ll monitor and answer them when I see them
Personnel

- Instructor
  - Geoff Voelker

- TAs and Tutors
  - Jiayou Guo (TA)
  - David Hacker (Tutor)
  - Hannah Hsu (Tutor)
  - Evan Laufer (Tutor)
  - Yingzhen Qu (TA)
  - Ana Selvaraj (Tutor)
  - Tianyi Shan (TA)
  - Mingyao Shen (TA)
  - Priyal Suneja (Tutor)
CSE 120 Class Overview

• Course material taught through class lectures, textbook readings, and handouts
  ♦ Starting with Lecture 2, I will post slides the day before class

• Course assignments are
  ♦ Homework questions
  ♦ Three large programming projects in groups
  ♦ Midterm and final exams

• Discussion sections
  ♦ Lecture material, homework, projects

• Other forums
  ♦ Piazza
Textbook

Homeworks

• There will be 4 homeworks throughout the quarter
  ♦ Reinforce lecture material

• Homeworks provide practice learning the material
  ♦ Unfortunately, wasted a lot of time and energy dealing with homework cheating in the past
  ♦ So: You get full credit for a technical answer related to the homework question
  ♦ Amount learned from doing homework is proportional to effort
  ♦ Your choice on how much effort
Nachos Project

DOCTOR FUN

"This is the planet where nachos rule."
Nachos

• Nachos is an instructional operating system
  ♦ It is a user-level operating system and a machine simulator
    » Not unlike the Java runtime environment
    » Will become more clear very soon
  ♦ Programming environment will be Java on Unix (Linux)
  ♦ The projects will require serious time commitments
    » Waiting until the last minute is not a good strategy

• You will do three+ projects using Nachos
  ♦ Concurrency and synchronization
  ♦ System calls, processes, multiprogramming
  ♦ Virtual memory

• You will work in groups of 1-3 on the projects
  ♦ Start thinking about partners
Exams

• Midterm
  ♦ Thursday November 5\textsuperscript{th} (put in your calendar)
  ♦ Covers first half of class

• Final
  ♦ Tuesday December 15\textsuperscript{th} (put in your calendar)
  ♦ Covers second half of class + selected material from first part
    » I will be explicit about the material covered

• Two exam periods
  ♦ 8am and 8pm (Pacific)
  ♦ Will be online for both in case there are questions

• No makeup exams
  ♦ Everyone must be able to attend these exam dates
    » Unless absolute dire circumstances
Grading

• Breakdown
  ♦ Homeworks: 6%
  ♦ Midterm: 28%
  ♦ Final: 33%
  ♦ Projects: 33%

• Course grades will be on a curve
• Do the work → Pass the class
  ♦ Academic integrity main reason students do not pass
Many Ways to Interact

- Lecture
- Discussion
  ✷ Thu 10-10:50am
- Office hours
  ✷ Mon 3-4pm & Wed 4-5pm
  ✷ All topics (lecture, project, hw, random, …)
- Lab hours
  ✷ TAs and tutors will have many lab hours
  ✷ For projects, but also anything else in the course
- Normally I’m in the labs the week projects are due
  ✷ Will have to do it online this quarter
Advice

• Watch the lectures
  ♦ Lecture material is the basis for exams and directly relates to the projects

• Do the homework
  ♦ Concepts seem straightforward…until you apply them
  ♦ Excellent practice for the exams, and some homework problems are exercises for helping with the project

• Ask questions
  ♦ Asking questions is the best way to clarify lecture material at the time it is being presented
  ♦ Piazza, lab + office hours will help with projects, homework
More Advice

• **Do not violate academic integrity**
  ♦ It is much better to get a 0 for an assignment than to fail the course for academic integrity violations
  ♦ If you are starting to panic – for any reason – contact me so that we can figure out a path forward
    » Especially given how crazy 2020 has been!

• **Start the projects early**
  ♦ They take longer than you might expect (really!)
Project 1 Scores

SCORE

START DATE

DAY.0-2  |  DAY.2-4  |  DAY.4-6  |  DAY.6-8  |  DAY.8-10  |  DAY.10-12  |  DAY.12-14  |  DAY.14-16

mean

median
Class Web Page

http://cseweb.ucsd.edu/classes/fa20/cse120-a/

• Serves many roles…
  ♦ Course syllabus and schedule (updated over quarter)
  ♦ Lecture slides
  ♦ Homework handouts
  ♦ Project handouts

• Optional material
  ♦ Entirely for your interest only

• Supplemental readings on Unix, monitors, and threads
  ♦ e.g., seminal research paper describing the early Unix system
  ♦ Concepts in paper might seem obvious and familiar, but they were new at one time
Recordings

- We will record lectures and discussion section
- Available via canvas
- Not unlike podcasting…
Questions

• Before we start on material, any questions about the class structure, contents, etc.?
Why?

You have a question, Calvin?

Yes! What assurance do I have that this education is adequately preparing me for the 21st century?

Am I getting the skills I'll need to effectively compete in a tough, global economy? I want a high-paying job when I get out of here! I want opportunity!
Why Operating Systems?

• Why take a course in operating systems?
  - It’s not like everyone will become OS developers, after all

• Understand what you use
  - Understanding how an OS works helps you develop apps
  - System functionality, performance, efficiency, etc.

• Pervasive abstractions
  - Concurrency: Threads and synchronization are common modern programming abstractions (Java, C#, C++, Rust, etc.)

• Complex software systems
  - Many of you will go on to work on large software projects
  - OSes serve as examples of complex systems
• This course addresses classic OS concepts
  ♦ Services provided by the OS
  ♦ OS implementation on modern hardware
  ♦ Interaction of hardware and software
  ♦ Techniques for implementing software systems that are
    » Large and complex
    » Long-lived and evolving
    » Concurrent
    » Performance-critical

• System software tends to be mysterious
  ♦ Virtual memory? Wazzat?

• Our goal is to explain those mysteries
```
# Top output

Top - 20:48:08 up 275 days,  1 user,  load average:  0.06,  0.07,  0.05
Tasks: 171 total,   1 running,  19 stopped,   0 zombie
Cpu(s):  0.1%us,  0.1%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem: 16467276k total, 1415960k used, 230k free
Swap:  0k total,   0k used,  884340k cached

PID USER PR NI VIRT RES SHMR S %CPU %MEM TIME+ COMMAND
14677 voelker  20  0  55548 3232  2364 R  0:00:07 top
24637 voelker  20  0  86300 6364 1024 S  32:06.70 mosh-server
  1 root  20  0  57812 1636  584 S  1:26.73 init
  2 root  20  0      0      0 0 0 0:03.13 kthreadd
  3 root RT  0      0      0      0 0 0:04.38 migration/0
  4 root RT  0      0      0      0 0 0:04.38 migration/1
  5 root RT  0      0      0      0 0 0:04.38 migration/1
  6 root RT  0      0      0      0 0 0:04.38 migration/1
  7 root RT  0      0      0      0 0 0:04.38 migration/1
  8 root RT  0      0      0      0 0 0:04.38 migration/1
  9 root RT  0      0      0      0 0 0:04.38 migration/1
 10 root RT  0      0      0      0 0 0:04.38 migration/1
 11 root RT  0      0      0      0 0 0:04.38 migration/1
 12 root RT  0      0      0      0 0 0:04.38 migration/1
 13 root RT  0      0      0      0 0 0:04.38 migration/1
 14 root RT  0      0      0      0 0 0:04.38 migration/1
 15 root RT  0      0      0      0 0 0:04.38 migration/1
```
Fundamental OS Concepts

• Processes and threads
  ♦ What they are, why we have them, how to implement them

• How to implement correct concurrent programs
  ♦ Synchronization, locks, condition variables
  ♦ Multithreaded applications

• Virtual memory
  ♦ What it is, why we have it, how to make it work

• File systems
  ♦ How to make persistent storage friendly to users and applications

• Some advanced topics at the end
What is an Operating System?

• How would you answer?
  ♦ (Yes, I know that’s why you’re taking the course…)
  ♦ (Note: There are many answers…)
What is an Operating System?

• The operating system is the software layer between user applications and the hardware

• The OS is “all the code that you didn’t have to write” to implement your application
The OS and Hardware

- The OS *abstracts/controls/mediates* access to hardware resources
  - Computation (CPUs)
  - Volatile storage (memory) and persistent storage (disk, etc.)
  - Communication (network, modem, etc.)
  - Input/output devices (keyboard, display, printer, camera, etc.)

- The OS defines a set of logical resources (*objects*) and a set of well-defined operations on those objects (*interfaces*)
  - Physical resources (CPU and memory)
  - Logical resources (files, programs, names)
  - Sounds like OO…
The OS and Hardware (2)

- Benefits to applications
  - Simpler (no tweaking device registers)
  - Device independent (all network cards look the same)
  - Portable (across Win95/98/ME/NT/2000/XP/Vista/7/8/10/…)
  - Transportable (same program across different OSes (Javascript))
The OS and Applications

• The OS defines a logical, well-defined environment…
  ♦ Virtual machine (each program thinks it owns the computer)
• …for users and programs to safely coexist, cooperate, share resources
  ♦ Concurrent execution of multiple programs (timeslicing)
  ♦ Communication among multiple programs (pipes, cut & paste)
  ♦ Shared implementations of common facilities
    » No need to implement the file system more than once
  ♦ Mechanisms and policies to manage/share/protect resources
    » File permissions (mechanism) and groups (policies)
More Questions to Ponder

• What is part of an OS? What is not?
  ♦ Is the windowing system part of an OS?
  ♦ Is the Web browser part of an OS?
More Questions to Ponder

• What is part of an OS? What is not?
  ♦ Is the windowing system part of an OS?
  ♦ Is the Web browser part of an OS?

• Popular OSes today are Windows, Linux, and OS X
  ♦ How different/similar do you think these OSes are?
  ♦ How would you go about answering that question?
More Questions to Ponder

- What is part of an OS? What is not?
  - Is the windowing system part of an OS?
  - Is the Web browser part of an OS?

- Popular OSes today are Windows, Linux, and OS X
  - How different/similar do you think these OSes are?
  - How would you go about answering that question?

- OSes change all of the time
  - Consider the series of releases of Windows, Linux, OS X…
  - What are the drivers of OS change?
  - What are the most compelling issues facing OSes today?
Pondering Cont’d

• How many lines of code in an OS?
  ♦ Win7 (2009): 40M
  ♦ OS X (2006): 86M
  ♦ Linux (2011): 15M
  ♦ What is largest kernel component?

• What does this mean (for you)?
  ♦ OSes are useful for learning about software complexity
  ♦ OS is just one example of many complex software systems
    » Chrome (2015): 17M
    » Hadoop (2018): 3.9M
    » JDK (2015): 6M
    » Unreal Engine 4: 2.3M
  ♦ As a software developer, you will face complexity
For next class...

• Browse the course web
  https://cseweb.ucsd.edu/classes/fa20/cse120-a/
• Sign up on Piazza!
• Read Chapters 1 and 2
• Start thinking about partners for project groups
• I will stay on zoom to answer questions
  ♦ And will disable the recording…