CS 45: Operating Systems
Course Introduction

Kevin Webb
Swarthmore College
January 23, 2018

Please sit towards the front, next to other students!
Instructor: Kevin Webb


• Please call me Kevin (or Professor Webb)

• Research: Control platforms for networks and distributed systems
Instructor: Kevin Webb


- Please call me Kevin (or Professor Webb)

- Research: Control platforms for networks and distributed systems

- Hobbies: Building stuff, cactus/fruit plants, PC games, weight lifting
Instructor: Kevin Webb

- http://www.cs.swarthmore.edu/~kwebb/
- Please call me Kevin (or Professor Webb)
- Research: Control platforms for networks and distributed systems
- Hobbies: Building stuff, cactus/fruit plants, PC games, weightlifting
Office Hours

• Tuesday 4:00 PM – 5:00 PM
• Friday 12:30 PM – 2:00 PM
• By appointment
• 255 Science Center
Resources

• Piazza Q&A Forum, GitHub Enterprise
  – https://github.swarthmore.edu

• Slides & audio recordings on course website

• Lab sections:
  – Science Center 240
  – Monday 1:15-2:45, Monday 3:00-4:30
Email Policy

• For public questions: use Piazza!
  – Your classmates benefit from your questions
  – Your classmates can answer your questions
  – I will check the forum frequently

• For private questions: use email (kwebb@cs)

• I will attempt to respond to within 24 hours
How does this class work?

- This class is designed a bit differently from what you might normally be used to
  - Class will be centered around you
  - Requires your participation

- Ever considered why we have lectures?
Traditional Lectures:

- Roughly one millennium old
Traditional Lectures:

- Little opportunity for expert feedback
- Might as well skip class and watch video lectures!
  - (I am not actually suggesting this. Please attend your classes!)
Interactive Classes with Peer Instruction

• You do the “easy” part before class.

Textbook, videos, website

First Exposure

In class quiz

Gauge understanding

Instruction

Fill in gaps, Explore details, Add context, Provide feedback

Exam

Show Knowledge Mastery

• Class is reserved for interactive, customized experiences

• Research on how people learn:
  – Everyone constructs their own understanding
  – To learn, YOU must actively work with a problem and construct your own understanding of it
Clickers!

- Lets you vote on questions in real time.
- Like pub trivia, but the subject is always networks.
- You NEED one of these for the course!
Peer Instruction

• **Short quiz at the beginning of class**

• **During class: pose carefully designed questions**
  – Solo vote: Think for yourself and select answer
  – Discuss: Analyze problem in teams of 3
    • Practice analyzing, talking about challenging concepts
    • Reach consensus
    • If you have questions, raise your hand and I’ll come over
  – **Group vote: Everyone in group votes**
    • You must all vote the same to get your point

– **Class wide discussion:**
  • Led by YOU (students) – tell us what you talked about in discussion that everyone should know!
Why Peer Instruction?

• You get a chance to think.
• I get feedback as to what you understand.
• It’s less boring!
• Research shows it promotes more learning than traditional lecture.
Giving out Candy

• To people willing to
  – Ask a question
  – Share an explanation
  – Summarize what their group talked about

• Your explanations are CRITICAL for fellow students’ learning
Example Question

• Individual vote

• Group discussion / group vote
  — Room should be LOUD

• Class discussion
Whose “magic” would be best to have? Why?

E: Some other character (be prepared to discuss)
Grading

• 5% Reading Quizzes
• 5% Class participation
• 25% Midterm Exam
• 30% Final Exam
• 35% Programming Assignments
Grading

• 0% Reading Quizzes
• 7.5% Class participation
• 25% Midterm Exam
• 30% Final Exam
• 37.5% Programming Assignments
Grading

• 0% Reading Quizzes
• 7.5% Class participation
• 25% Midterm Exam
• 30% Final Exam
• 37.5% Programming Assignments

• I will drop three no-shows. Course website has (CS department standard) extended absence policy.
Grading Scale

• A+: 99+
• A:  93-99
• A-: 90-93
• B+: 87-90
• B:  83-85
• B-: 80-83
• C+: 77-80
• ...

...
Textbook

- By Silberschatz, Galvin, and Gagne

“Dinosaur book”
Course Policies

• Collaboration
  – You may discuss approaches, not solutions
  – You must submit your own work
  – Exams will include questions on programming

• Cheating
  – Zero tolerance for cheating, don’t do it!

• Lab Lateness
  – 2 days of extra (at the granularity of days)
  – Let me know after you’ve submitted
Tentative Schedule

• Midterm – March 8, in class

• Final - TBD

• Labs
  – Released on Mondays (lab section)
  – Due on Sundays
  – Multi-week labs. Start early!
Labs

• This will be a lab-heavy course with lots of C programming! When I say “start early”, I mean work a little and then let it stew...

• “Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it.” -Kernighan’s Law

• Course content will not always match labs. Sorry! This is to give you more time on the labs...
Administrative Questions?

- All of this info (should be) on class website
- Feel free to ask on Piazza discussion board
Your TODO List

• Sign up on Piazza!

• Register your clicker!
  – https://cs.swarthmore.edu/clickers/

• Please let me know ASAP about:
  – Your preferred name/pronouns, if different than roster information
  – Academic accommodations
What *is* an Operating System?

- **Textbook:** “A program that … provides a basis for application programs and acts as an intermediary between the computer user and computer hardware.”

- **Wikipedia:** “System software that manages computer hardware and software resources and provides common services for computer programs.”
What *is* an Operating System?

- Textbook: “A program that ... provides a basis for application programs and acts as an intermediary between the computer user and computer hardware.”

- Wikipedia: “System software that manages computer hardware and software resources and provides common services for computer programs.”
What *is* an Operating System?

- **Textbook:** “A program that ... provides a basis for application programs and acts as an intermediary between the computer user and computer hardware.”

- **Wikipedia:** “System software that manages computer hardware and software resources and provides common services for computer programs.”
What is an Operating System?

For this course, “Operating System” refers to *just the OS Kernel*.

That is, just the low-level control software, not all userspace utilities!
What this Course is NOT

• Deep dive into details of one particular OS
  – We will use / modify Linux in the lab though!

• System administration / configuration
Major Course Themes

1. Abstractions (Hardware and Software)

Hiding complex details of reality behind a simpler illusion or interface.
Abstraction

- User / Programmer
  - Wants low complexity

- Applications
  - Specific functionality

- Software library
  - Reusable functionality

- Operating system
  - Manage resources

- Complex devices
  - Compute & I/O
Abstraction

User / Programmer
Wants low complexity

Applications
Specific functionality

Software library
Reusable functionality

Operating system
Manage resources

Provide abstract services!

Hide the details of complex hardware from the software above.
OS Abstractions

• Primary: Process (program in execution)

• Processes need resources...
  – Threads of execution
  – Virtual memory address space
  – Files (and file-like things: sockets, pipes, devices)

• Interfaces for user control
  – e.g., open/read/write/close for files
Major Course Themes

1. Abstractions (Hardware and Software)
2. Hardware Gatekeeping and Protection

OS enforces control over which processes/users can access resources.
Gatekeeping and Protection

• Clearly defined user process < -- > OS interface

• On any system call, verify permission and validate parameters

• Potentially limit access to resources (e.g., memory, files)
Major Course Themes

1. Abstractions (Hardware and Software)
2. Hardware Gatekeeping and Protection
3. Resource Sharing and Multiplexing

Multiple processes share the machine. How much CPU time should they get? Memory? Can/should one be prioritized?
### Reality
- Multiple processes
- Small number of CPUs
- Finite memory

### Abstraction
- Process is all alone
- Process is always running
- Process has all the memory
Major Course Themes

1. Abstractions (Hardware and Software)
2. Hardware Gatekeeping and Protection
3. Resource Sharing and Multiplexing
4. Design Decisions and Tradeoffs

The “best” solution typically depends on the scenario and expected use case.
Design Tradeoffs

• Design decisions for super computer are very different from mobile phones...

• This course: mainly focus on general-purpose computing (desktops, laptops), especially in lab assignments

• During discussions: consider alternative platforms (embedded devices, real-time devices, super computers, etc.)
Major Course Themes

1. Abstractions (Hardware and Software)
2. Hardware Gatekeeping and Protection
3. Resource Sharing and Multiplexing
4. Design Decisions and Tradeoffs
5. Mechanism vs. Policy

What we can do (and how) as opposed to what we should do (governing)
Policy vs. Mechanism (by example)

• Mechanism: processes can request memory, OS can validate and choose to satisfy request or not.

• Policy: admins get as much memory as they want, regular users are limited to X GB
Why this material is important...

• Systems is core to the business model of many HUGE employers... Google, MS, Amazon, etc.

• To be the best programmer you can be, you MUST understand what’s happening at the next layer down!
Reminder: Your TODO List

• Sign up on Piazza!

• Register your clicker!
  – https://cs.swarthmore.edu/clickers/

• Please let me know ASAP about:
  – Your preferred name/pronouns, if different than roster information
  – Academic accommodations