What is an Operating System?

June 21st, 2021
Neil Kulkarni and Akshat J. Gokhale
https://cs162.org
Goals for Today

• What is an Operating System?
• Oh, and “How does this class operate?”
What is an operating system?
Definition of an Operating System

- No universally accepted definition
- “Everything a vendor ships when you order an operating system” is good approximation
  - But varies wildly

- “The one program running at all times on the computer” is the kernel
  - Everything else is either a system program (ships with the operating system) or an application program
One Definition of an Operating System

• Special layer of software that provides application software access to hardware resources
  – Convenient abstraction of complex hardware devices
  – Protected access to shared resources
  – Security and authentication
  – Communication
What you learned in CS 61C – Machine Structures (and C)

The OS *abstracts* these hardware details from the application.
Operating Systems are at the Heart of Everything!

• Make incredible advances in technology available to a rapidly evolving body of applications
  – Provide **consistent abstractions** to applications, even on different hardware
  – Manage **sharing of resources** among multiple applications

• The key building blocks:
  – Processes
  – Threads, Concurrency, Scheduling, Coordination
  – Address Spaces
  – Protection, Isolation, Sharing, Security
  – Communication, Protocols
  – Persistent storage, transactions, consistency, resilience
  – Interfaces to all devices
Questions

• Does the programmer need to write a single program that performs many independent activities?
• Does every program have to be altered for every piece of hardware?
• Does a faulty program crash everything?
• Does every program have access to all hardware?

Hopefully, no!

Operating Systems help the programmer write robust programs!
What are the roles of an Operating System?

- **Referee**
  - Manage sharing of resources, Protection, Isolation
    - Resource allocation, isolation, communication

- **Illusionist**
  - Provide clean, easy to use abstractions of physical resources
    - Infinite memory, dedicated machine
    - Higher level objects: files, users, messages
    - Masking limitations, virtualization

- **Glue**
  - Common services
    - Storage, Window system, Networking
    - Sharing, Authorization
    - Look and feel
Basic Tool: Dual-Mode Operation

• Processor provides at least two modes:
  1. Kernel Mode (or “supervisor” mode)
  2. User Mode

• Certain operations are **prohibited** when running in user mode
  – Disabling interrupts, interacting directly with hardware, writing to kernel memory

• Carefully controlled transitions between user mode and kernel mode
  – System calls, interrupts, exceptions
“So…Operating Systems:”

- Provide convenient abstractions to handle diverse hardware
  - Convenience, protection, reliability obtained in creating the illusion
- Coordinate resources and protect users from each other
  - Using a few critical hardware mechanisms
- Simplify application development by providing standard services
- Provide fault containment, fault tolerance, and fault recovery

- CS162 combines things from many other areas of computer science:
  - Languages, data structures, hardware, and algorithms
Why take CS162?

• Some of you will actually design and build operating systems or components of them.
  – Perhaps more now than ever
• Many of you will create systems that utilize the core concepts in operating systems.
  – The concepts and design patterns appear at many levels
• All of you will build applications, etc. that utilize operating systems
  – The better you understand their design and implementation, the better use you’ll make of them.
Intros - Akshat J. Gokhale

• Graduated UCB in Spring 2021
  – B.S. in Electrical Engineering & Computer Science
  – B.S. in Business Administration

• Teaching experience:
  – CSM Junior Mentor (CS61C SP19)
  – TA (CS162 SP20)
  – Head TA (CS162 FA20 + SP21)

• Post-grad plans:
  – Business Analyst (McKinsey & Co.)
• Graduated UCB in Spring 2021
  – B.S. in Computer Science
  – B.S. in Applied Mathematics

• Teaching experience:
  – **CS 61A TA** (FA18)
  – **CS 61B TA** (SP19 + SP21)
  – **CS 61C TA** (FA19)
  – **CS 162 TA** (SP20)
  – **CS 162 Head TA** (FA20)

• Post-grad plans:
  – FPGA Engineer (Citadel)
  – Grad School?
CS162 TAs

Edward Zeng  
(Head TA)

Animesh Agarwal

Rahul Kumar

Sean Kim
Enrollment

• Class has 90 limit
  – Only 80 enrolled, tell your friends to enroll!

• Drop deadline is July 2nd
  – If you are not serious about taking the course, please drop early
  – Department will continue to admit students as other students drop
  – Really hard to drop afterwards (financial penalty!)
Prerequisites

• The official prerequisites for this class are CS 61A, CS 61B, CS 61C, CS 70
  • Data structures: arrays, linked lists, binary trees, and hashing
  • Assembly language programming
  • The C programming language
  • Debugging C using GDB
  • CPU caches and memory hierarchy
  • Virtual memory as covered in CS 61C
  • CPU pipelines and basic digital logic design
  • Basic knowledge of random variables and probability distributions as covered in CS 70

• We will not spend any time in lecture covering this material
  – We will assume that you either know the material that is supposed to be covered in those courses, or that you are willing to learn the material as necessary
  – The TAs will spend a small amount of time reviewing some of the material you are less likely to remember

• If you have taken similar prerequisite courses at other universities, that is ok!
• Perhaps more important than formal prerequisites, however, is experience and maturity with debugging large programs, designing and implementing useful abstractions, and computational problem solving in general.
Infrastructure, Textbook & Readings

• Infrastructure
  – Website: https://cs162.org
  – Piazza: https://piazza.com/berkeley/summer2021/cs162

• Textbook: Operating Systems: Principles and Practice (2\textsuperscript{nd} Edition)
  Anderson and Dahlin
  – Required textbook readings posted along with lectures
  – Try to keep up with material in book at the same pace as lectures

• Supplementary Material
  – Operating Systems: Three Easy Pieces, by Remzi and Andrea Arpaci-Dusseau,
    available for free online
  – Linux Kernel Development, 3\textsuperscript{rd} edition, by Robert Love

• Online supplements
  – See course website
  – Includes appendices, sample problems, previous exams, etc.
  – Networking, Databases, Software Engineering, Security
  – Some Research Papers!
Syllabus

• OS Concepts
  – Processes, I/O, Virtual Machines
• Concurrency
  – Threads, Scheduling, Locks, Deadlock, Scalability, Fairness
• Address Space
  – Virtual Memory, Address Translation, Protection, Sharing
• File Systems
  – I/O Devices, File Objects, Storage, Naming, Caching, Performance, Paging, Transactions
• Distributed Systems
  – Networking, RPC, Distributed Storage
Lecture, Instructor OH

• Lecture
  – Lectures 1-23 will be released as recordings
    » Each lecture will be released one day prior to the scheduled lecture time
    » Scheduled lecture time will be reserved for a lecture Q&A
  – Lectures 0 and 24 (the first and final lectures) are live!
    » Recordings will be posted the day after
  – (Optional) Special Topics Lectures (Lectures 25-26) will also be live!
    » If you have ideas on topics you would like to hear about, let us know!
    » Topics in the past include: blockchain, quantum computing, security, etc.
  – Make sure to sign in to Zoom using SSO (details will be posted on Piazza)

• Instructor OH
  – We will be hosting conceptual OH on Fridays by appointment (20m)
    » Signup links can be found on Piazza and the website OH calendar
  – Akshat: 9am-1pm PST
  – Neil: 1pm-5pm PST
TA/Reader OH

• TAs and Readers will host OH throughout the week
  – OH will always be hosted in the same Zoom room

• For collaboration with other students:
  – Use the public breakout rooms to discuss homeworks/projects with other students
  – Screensharing or code sharing (even verbally) are NOT permitted

• For conceptual or debugging help from course staff:
  – Create a request on the OH queue (see link on the course website)
  – When a staff member picks up your ticket, they will invite you to a private breakout room
  – You MUST follow the ticket template that is provided, or else your ticket may be skipped
Learning by Doing

• Individual Homeworks (~1 week each)
  – HW0. Tools & Environment, C Review
  – HW1. Lists in C
  – HW2. BYOS (Build Your Own Shell)
  – HW3. Synchronization
  – HW4. Sockets & Threads in HTTP Server
  – HW5. Memory Allocation (malloc)
  – HW6. Memory Management

• Three (and ½) Projects
  – Proj0. Getting Started (Individual, before you have a group)
  – Proj1. User Programs & Syscalls
  – Proj2. Threads & Scheduling
  – Proj3. File Systems
Group Projects

• Project teams have 4 members!
  – Never 5, groups of 3 only if class is not divisible by 4
  – Must work in groups in “the real world”
  – Same section and same TA
  – If you don’t have a group, see Piazza for a group finding thread OR come to the Zoom Student Mixer on Thursday 6-7 PM PT

• Communication and cooperation will be essential
  – Regular meetings WITH CAMERA TURNED ON!
    » Extra credit for screen shots of all of you together in Zoom with cameras on
  – Design Documents
  – Slack/Messenger/whatever doesn’t replace face-to-face!

• Everyone should do work and have clear responsibilities
  – You will evaluate your teammates at the end of each project
  – Dividing up by task is the worst approach. Work as a team!

• Communicate with supervisor (TAs)
  – What is the team’s plan?
  – What is each member’s responsibility?
  – Design Documents: High-level description for a manager!
Getting started

• Pre-Semester/Timezone Survey! Due Friday, June 25th
  – We need to know where you are so that we can plan section/midterm times
• If you are not enrolled in Gradescope or the Autograder, post on Piazza!
• Start HW0 right away (translation: today!)
  – Register on the autograder by Thursday, June 24th
  – Vagrant virtualbox – VM environment for the course
    » Consistent, managed environment on your machine
    » Start now in case there are any setup issues with your VM!
  – Get familiar with all the CS162 tools and brush up on C
  – Submit to autograder via git
• Start Proj0 as soon as it is released later this week
• Sections on Wednesday and Friday – attend any section you want this week
  – We’ll assign permanent sections and finalize project groups this weekend
  – Section attendance will be mandatory starting next week
  – Current section times will be adjusted after we have a better idea where people are
Preparing Yourself for this Class

• The projects will require you to be very comfortable with programming and debugging C
  – Pointers (including function pointers, void*)
  – Memory Management (malloc, free, stack vs heap)
  – Debugging with GDB
• You will be working on a larger, more sophisticated code base than anything you've likely seen in 61C!
• Review Session on C:
  – Tuesday, June 22nd (6-8PM PST)
  – More information on Piazza, recording will be posted
• "Resources" page on course website
  – Ebooks on “git” and “C”
• C programming reference (still in beta):
  – https://cs162.org/ladder/
• First two sections will cover programming and debugging review:
  – Attend ANY sections during first week
Grading Breakdown

- **18% Homeworks** (8 total, 2.25% each)
  - Includes Proj0
  - Submission via *git push* triggers autograder

- **36% Projects** (3 total, 12% each)
  - Submission via *git push* triggers autograder
Grading Breakdown

• **36% Exams**
  – **Midterm:** Friday, 7/16, time set after pre-semester survey
  – **Final:** Tuesday, 8/10, time set after pre-semester survey
  – These will be Zoom-proctored. Camera REQUIRED.
  – Final is cumulative
  – Thoroughly understanding lectures, discussions, the textbook, and projects are instrumental in performing well on exams in this course

• **Clobber Policy**
  – **Default:** 16% Midterm, 20% Final
  – **Final > Midterm:** 12% Midterm, 24% Final
Grading Breakdown

- **10% Participation**
  - 35% Design Review Participation
    » All design reviews are mandatory
  - 20% Discussion Attendance
    » Mandatory starting Week 2
    » Notify your TA beforehand if you cannot attend, and you may be excused
  - 15% Discussion Participation
    » Speak up! (verbally or in the Zoom chat)
  - 20% Weekly Lecture Quizzes
    » Delivered during discussion
    » 80% or more is a full score
    » If you are caught up on lecture, these will be extremely easy
  - 10% Mandatory Surveys
    » Posted on Piazza and the course website
Late Policy

- **All assignments are due at 11:59 PM PST on the day listed on the course calendar**
- **Slip Days**
  - You can use each slip day to extend a deadline by exactly 24hrs.
    » This will be applied automatically if you submit an assignment after the deadline
  - You get 2 slip days for homework assignments
    » Project 0 will use HW slip days (individual assignment)
  - Your group gets 3 slip days for group assignments
    » Project slip days cannot be used on design documents
- **If you are stressed about turning in an assignment on time, please contact your TA**, please do not resort to academic dishonesty!
Personal Integrity

- UCB Academic Honor Code: "As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others."

https://asuc.org/honor-code-landing/
CS 162 Collaboration Policy

Explaining a concept to someone in another group
Discussing algorithms/testing strategies with other groups
Discussing debugging approaches with other groups
Searching online for generic algorithms (e.g., hash table)

Sharing code or test cases with another group
Copying OR reading another group’s code or test cases
Copying OR reading online code or test cases from prior years
Helping someone in another group to debug their code

• READ THE COLLABORATION POLICY ON THE WEBSITE IN ITS ENTIRETY
  • We compare all project submissions against prior year submissions and online solutions and will take actions (described on the course overview page) against offenders
  • Don’t put a friend in a bad position by asking for help that they shouldn’t give!
Academic Dishonesty

• Dr. Nick Weaver will be managing all academic dishonesty cases for this course this term along with us
  – Weaver is a computer security lecturer at Berkeley

• DO NOT mess with him! He will catch you!
  – He inscribes a new notch on his wooden staff for every cheater that he catches… scary stuff.

YOU SHALL NOT PASS
(if you cheat)
CS162 in the age of COVID-19

• We'd like to not over-stress you too much
  – There really is a lot to cover and this really is a demanding class
  – But, we're here to help!

• If you feel overwhelmed, please use the resources available
  – Academically: Ask on Piazza, Slack, OH, Instructor/TA Email
  – Non-Academic: Take advantage of University Health Services if you need to

• There are things that are out of your control
  – If something bad happens near the end of the semester, there are withdrawals and incompletes
CS162 in the age of COVID-19
CS162 in the age of COVID-19

• Well, things are (still) virtual this term!
  – Many lessons learned from four virtual semesters
  – Everything is remote – all term!

• Most important thing: People, Interactions, Collaboration
  – How do we recover collaboration without direct interaction?
  – Remember group meetings?

• Must *work* to bring everyone along (virtually)!
  – Cameras are *essential* components of this class
    » Must have a camera and plan to turn it on
    » Will need it for exams, discussion sections, design reviews, OH
  – Required attendance at: Discussion sections, Design Reviews
    » With camera turned on!
  – Need to bring back personal interaction – even if it is virtual...
Welcome to LogPorts Castle!

• Next week, the **Scheduling Hat** will decide your TA (and your LogPorts house!)
  – Edward, Head of Gryffindor
  – Rahul, Head of Ravenclaw
  – Animesh, Head of Hufflepuff
  – Sean, Head of Slytherin

• Throughout the semester, you and your house will compete for points
  – Competition details will be released on Piazza and announced during discussion

• The winning team will:
  – Receive a small EC boost (after the curve)
  – AND feel free to brainstorm additional prizes on Piazza!
    » (e.g. winning team gets to choose a prank to play on the instructors???)