Administrivia: Course Organization (1)

• Pre-class work
  • First presentation of material
  • Reading and videos that introduce new concepts.
  • Short questions to demonstrate that you got the big picture.

• Lecture: Practice and guided exploration
  • Group exercises that probe the concepts in more depth.
  • Design something.
  • Code something.
  • Measure something.
  • Test something.
Administrivia: Course Organization (2)

- Section: Details -- especially on assignments
  - Connect class exercises to assignments
  - Suggestions and discussion about the assignments
  - Exam preparation
  - For those of you who don’t want/need section: we will offer one “special topics” section, which will not discuss assignments, but will cover interesting topics that apply concepts learned in class in different environments.

- At the beginning of the semester, we will also offer some sections to help you become more comfortable with some of the tools that we will use in class: git, gdb, the shell, etc.
Administrivia: Assignments (1)

- Six assignments
- Each is approximately two weeks
- Most may be completed in pairs
- Many have an “intermediate” checkin (required).
- Each assignment gives you hands-on experience with a key concept in systems programming.
- Assignments are graded on both correctness and style.
- You have a total of 144 late hours.
Administrivia: Assignments (2)

• A1: A Debugging Memory Allocator
  • Become expert at manipulating memory.
  • Develop tools that help you write more robust code.

• A2: Binary Bomb
  • Become an expert debugger, digging into assembly language in gdb.
  • Read and understand x86 assembler.

• A3: Add Caching to a Standard IO package
  • Use caching to improve system performance.
  • Unleash the magic hidden in standard libraries.
Administrivia: Assignments (3)

• A4: Write your own shell
  • Become proficient at manipulating processes.
  • Internalize that a program is a program is a program, even if it’s a shell.

• A5: Solve some synchronization problems
  • Select appropriate synchronization primitives to solve the problems that arise from concurrency.
  • Become comfortable with multi-threaded programming.

• A6: Write a virtual memory system for Weensy OS
  • Discover how much you’ve learned about operating systems.
  • Demonstrate how processes provide the abstract machine.
Administrivia: Participation

• Class is required (yes, we’ll take attendance).
• Participation includes:
  • Completing pre-class work
  • Actively participating in group exercises
  • Completing post-class surveys
Administrivia: Exams

• Midterm
  • In-class, 82 minute exam on October 22.
  • Allow you to demonstrate mastery of the concepts covered in the first half of the course.

• Final
  • 3-hour exam during exam period.
  • Allow you to demonstrate mastery of the concepts covered in the whole course, with emphasis on the second half.
  • Some of the questions will ask you to apply what you’ve learned to situations you have not encountered.