CSE 306
Operating Systems
Introduction
Don Porter

Paperwork
+ I am handing out a survey on your background and mini quiz
+ Please complete and return before you leave.

What is an OS?
+ All of the stuff between you/your application and the hardware
  + Kernel
  + Device Drivers
  + API libraries
  + UI
  + Our focus is mostly on the kernel, with some attention to the others

Why Operating Systems?
+ Primary Goal: Demystify how computers work
  + Lots of abstractions and heuristics between your application and the hardware
  + A good computer scientist should understand what happens inside the system when one types a command
+ Secondary: Learn how to write robust programs
  + OSes like Linux have many users and work on a wide range of hardware
  + Deal with subtle issues: concurrency, consistency, etc.

Labs: Learn by doing
+ This course is coding intensive
  + You should know C, or be prepared to remediate quickly
  + You will learn basic, inline x86 assembly
  + You must learn on your own/with lab partner
  + You will write substantial applications in C
  + Final project will involve substantial modifications to the Linux kernel
  + Challenging, but a very marketable skill

Lab Teams
+ Lab 1: Everyone does this lab alone
+ Lab 2 and 3: May work with a partner or alone
+ Lab 4: May work in a team up to 4 students
Lab Teams

- Can work alone, but better with help
- No need to be a hero
- Choose your own partners
- Course mailing list good for finding them
- Same for entire course
- Changes only with instructor permission
- For lab 4, you can only join with another team

Challenge Problems

- Each lab may include challenge problems, which you may complete for bonus points (generally 5—10 points out of 100)
- Unwise to turn in a lab late to do challenge problems
- Can complete challenge problems at any point in the semester—even on old labs
- Indicate any challenge problems completed in challenge.txt file

Administrative

- Syllabus, schedule, homework, etc. posted on course website
  
  www.cs.stonybrook.edu/~porter/courses/cse306/s15

Required Readings

- Primarily from the class textbook
- Should be completed before the lecture
- Required reading material may appear on the exams, even if not discussed in lecture
- Several recommended texts will be posted
  - Several free on SBU safari online site
  - Papers you can print out or read electronically
  - Others on reserve at library

Lectures

- Discuss and supplement reading material
- An important chance to clarify issues
  - Questions are encouraged!
  - I expect you to arrive prepared to answer and ask questions about the reading material
  - Everything in lectures may appear on the exams, even if not in the book

Prerequisites

- CSE 219 (CS III) or CSE 260 (CS II, Honors)
- CSE 220 (Systems-level Programming) or ESE 380 (Embedded Microprocessor Design I)
- The background courses are necessary
- In some cases, industry experience is ok
  - In-class quiz, due before you leave
    - If you can't answer 50% of these questions you are not prepared
- C programming
- Basic Unix command-line proficiency
Course email list
+ We will use Piazza this semester. Details will be posted on course website
+ This is the primary announcement medium
+ And for discussions about course work
+ Do not post code here or other solutions
+ Goal: Everyone can learn from general questions
+ Material discussed on the mailing list can be an exam question

Special Offer!
+ You can write your own exam questions
+ Send them to me in advance of the test, if I like them, I will use them
+ Do NOT share with anyone else

Why do we care?
+ Analogy: This is the programming dojo
+ If you don't do your exercises, you will be unprepared for battle
+ You've wasted your money and both of our time
+ It brings dishonor on the dojo when you lose every battle
+ Similarly, a lot of what I have to teach (and what will make you a valuable employee when you graduate) has no short cut
+ How do you learn to punch through a board?
+ You punch a board over and over until your fist goes through it
**Productive Frustration**

- One of the "meta skills" that distinguishes an excellent programmer is the ability to get un-stuck
- Fixing a "heisenbug" has this property
- How do you learn this skill?
  - Get stuck on a hard, but solvable problem
  - Learn which strategies will get you moving again
  - If you take a quick cheat, you won’t learn the skills to solve truly hard problems

**Lateness**

- Each student gets 72 late hours
- List how many you use in slack.txt
- Each day after these are gone costs a full letter grade on the assignment
- If you work in a team, each member loses 1 hour for each hour late
- It is your responsibility to use these to manage:
  - Holidays, weddings, research deadlines, conference travel, Buffy marathons, release of the next Zelda game, etc.
- 3 Exceptions: illness (need doctor's note), death in immediate family, accommodation for disability

**Getting help**

- TA's will keep office hours (TBD)
- Instructor keeps office hours
  - Note that “by appointment” means more time available on demand

**Questions?**

- Remember:
  - Hand-in survey
  - Assignment coming out soon
  - Reading assigned for Thursday