Course Overview

COS 316: Principles of Computer System Design

Lecture 2

Amit Levy & Wyatt Lloyd

Course Staff: Intros

Learning Objectives & Course Components

- System Design Principles
  - Lectures
  - Problem Sets
  - Final Project

- Skills
  - Precepts
  - Programming Assignments
  - Final Project

Learning Objectives: System Design Principles

- What is the field of systems?
  - Learn to appreciate trade-offs in designing and building the systems you use.
  - Get better at understanding how systems work.
  - Learn to *use* systems better---write more efficient/secure/robust/etc applications.

Lectures

- Attend synchronously (if possible)
  - Active thinking through concepts (you)
  - Active calibration of teaching (us)

- Explore fundamental concepts,
  ways of thinking,
  cutting-edge research

Lectures

- 6 Major Themes:
  - Naming
  - Caching
  - Layering
  - Concurrency
  - Access Control
  - Scheduling

Problem Sets

- Focus on reinforcing and generalizing lecture content
- Done individually

Learning Objectives: Skills

- Go programming language, and "Systems" programming
- Version control with git
- Working in groups
- "Systems programming": sockets programming, concurrency, modular design, unit testing, performance measurement, ...

Precepts

- Attend synchronously
  - Range of time zones covered, one will work for you!
- Hands on, active learning in small groups
- Coupled primarily with the programming assignments

Programming Assignments

- You're Building a Web Framework!

- Set of libraries and tools for building complex web applications
  - Abstracts connection and protocol handling
  - Routes requests to controllers/handlers
  - Caching for common queries and computations
  - Multiplexes concurrent access to databases
  - Translates database objects into programming language constructs
  - User authentication and authorization

- Examples: Rails, Django, Express, Apache Struts, Laravel

WARNING

Systems Building is *not just* Programming

- COS126 & 217 told you how to design & structure your programs.
  - This class doesn’t.
- If your system is designed poorly it can be much harder to get right!
- Conversely, assignments won't require algorithms or data structures you're not already familiar with.
  - 4xx systems classes require both!
- Your friends:
  - Working in teams (don't worry, you're *required to*)
  - Discussing potential solutions before implementing
  - Test-driven development

Assignments: Collaboration & Resources

This slide is really important

- You can, and *should* any resources available on the Internet to complete assignments:
  - Go documentation, Stackoverflow, open source projects
  - Mailing lists, chat rooms, etc...
  - Cite sources in your comments or README!
- You *must* collaborate (in groups of 2)
- You may *not* ask instructors for help debugging your code.
- ~~*Gilligan's Island* *Game of Thrones*~~ Take-a-walk rule:
  - If you discuss the assignment with other teams, do something else for an hour before returning to your code

Assignments: Collaboration & Resources

https://cos316.princeton.edu/assignments

activity	your group*	course staff	COS 316 grads	classmates	other
discuss concepts with ...	✔	✔	✔	✔	✔
acknowledge collaboration with ...	✔	✔	✔	✔	✔
expose solutions to ...	✔	✔	✘	✘	✘
view solutions from ...	✔	✘	✘	✘	✘
plagiarize code from ...	✘	✘	✘	✘	✘

Assignments: Submitting and Grading

- Submitting happens whenever you "push" to your "master" branch on GitHub
  - You can push as many times as you'd like (we encourage you to do so *often*)

- Grading is automatic and immediate
  - There is no penalty for multiple submissions. We will use your highest graded submission (push)
  - Each automatic grading is posted as a comment to the last commit of each push. It includes a break down of tests cases, including which failed.

Programming Assignment Late Days

- 7 late days total for the semester
  - Granularity of 1 day
    - 1102pm on Wednesday is 1 day late
    - 1050pm on Thursday is 1 day late

- Assigned retroactively to give you the best possible overall grade
  - We do this for you!

Late Days Example

1. Parker submits assignment #1 on time, but can't figure out how to pass the last test case. Their grade so far for the assignment is 95%.
2. 7 days after the deadline, Parker figures out how to pass the last test and submits late, getting 100%.
3. Months later... Parker underestimates their workload and isn't able to submit assignment 4 until 7 days after the deadline, but passes all tests to get 100%.

4. We assign the late days to assignment 4, so that Parker's grade is 95% + 100%, as opposed to 100% + 0%.

Final Project

- Open ended systems building project
- Later precepts will help you refine topic
- You design and build something you're interested in!
- Small written component (< 2 pages)

What is Due When?

- Alternating Problem Sets and Assignments each week
  - Each is due on Wednesday at 11pm Princeton Time

- Final project is due on Deans Date at 5pm Princeton Time

Grading

- 60% - Programming Assignments
  - 6 Assignment, each worth 10%
- 20% - Problem Sets
- 20% - Final Project
- No curve anticipated
  - Will **not** curve down (i.e., a 93% is an A no matter what)

Learning Objectives & Course Components

- System Design Principles
  - Lectures – Attend Synchronously
  - Problem Sets – Due every other week
  - Final Project – You build something new

- Skills
  - Precepts – Attend Synchronously
  - Programming Assignments – Due every other week
  - Final Project – Due on Dean's Date