Introduction

Class 0

Overview

1. Staff introductions
2. Class overview
3. Tips for success

Staff Introductions

IA: Zalan Shah

- CS-Eng
- Complex Systems minor
- Interests include open source, machine learning, and app development
- Hobbies include longboarding, rubiks cubes, and video games
- Fun fact: has gone without sleep for 4 days before

IA: Kethan Gudi

- CS-Eng
- Interests include mathematical models and data science
- Proud moped owner
- Enjoys cooking, jiu jitsu, volleyball, and tennis

Instructor: Brandon Nguyen

- PhD Student under Prof. Scott Mahlke
- Undergrad at UT Austin in ECE
  - Primary focus: Computer architecture and embedded systems
  - Secondary focus: Digital signal processing
  - (spent my first two years as a BME doing premed...)
- Interests include computer architecture, compilers, and systems software
  - Weirdo who enjoys classes like 427, [2345]70, [34]73, 482, 583

Random fun stuff about me

- Modern and historical fencing (HEMA)
- "Slight" addiction to Genshin Impact
- Finally gotten around to indulge in photography and videography
- Trying to learn Vietnam's pre-romanization writing system
  - Turns out knowing written Chinese is a prerequisite

Course Overview

What is this class

- This class is for *anyone* wanting to become more effective at using their computer for development work
- This isn't necessarily a "tools" class
    - Tools come and go: does anyone remember COBOL and CVS?
- Each workplace will have its own tools and workflows
- The ultimate goal of this class is to help you learn to pick up, learn, and use new tools to solve problems
- The tools you learn along the way are the icing on the cake
- That being said, we will be focusing on Unix/Unix-like systems and shells in this class
    - Windows Command Prompt is not suitable for this class
    - Windows Subsystem for Linux (WSL) is suitable, however

Expectations

- Have a basic understanding of program control flow
  - e.g. if statements, loops, functions
- Have experience expressing your solutions in program statements
- Have some experience with a C or C++ or similar language
  - Let me know if you need help with the language itself
- Work is intended to be done alone
  - It can help to point each other to useful resources you find
  - Your code should be your own

Expectations

Technology

- Have a computer that runs Windows, mac OS, or Linux that you can install software on
- Chromebooks are welcome if they have Linux Beta (Crostini)
- Ubuntu 20.04 is going to be the reference environment for class
  - Most Linux distributions will have similar behvaior for things in this class, so don't fret if you're on 22.04 or Debian or whatever $^{\text{i use arch btw}}$
- While most things can be done on mac OS, some tools have different behavior between Linux and mac OS

Course communication

- Canvas: Announcements and a fancy gradebook
- Piazza: Course content and logistics related questions
- Discord: Casual and informal chat and questions
- Email: For personalized correspondance and more personal matters
    - Please start the subject line with "EECS 201" so I can find it

Course structure

Flipped classroom

- "Lecture at home, homework at lecture"
- Weekly content videos
- Quick quiz over videos with time window (keep up with class!)
- In-person class, attendance optional
  - Mini-review
  - Q/A
  - Time to work on assignments with instructor around for questions!
  - Survey for extra credit

Course structure

Weekly "basic" assignment

- Guided light assignments to familiarize you with tools and what you can do with them
- Directly related to material covered in the videos

Advanced component

- Less guidance than basic assignments
- May touch on some things not covered in lecture
- Provides practical experience in perusing documentation and applying what you know
- Can also be fulfilled by doing a project
    - Checked out at an office hour
    - More to come about this...

Grading

- Point accumulation
- Two major grade categories: **Basic** and **Advanced**
- Basic has 60 total points
- Advanced has 40 total points
- Final score is the *adjusted* sum of these categories
    - You can make more than 60 Basic or 40 Advanced points
    - Points after the category total are worth half (more on this)
    - There is no averaging: you just add numbers
    - Video quiz and survey credit is added on top
    - You can see how letter grades get assigned in the syllabus

Grading

Basic

- There will be at least 10 basic assignments, worth 6 points each
- That means you only need to do 10 to get all 60 points
- The remaining assignments serve as a buffer for you to miss/skip
- **Points past 60 are worth 50%: an 11th assignment would only be worth 3 points**
- If you do 12 basic assignments:
  - 12 * 6 = 72 -> 60 + 12/2 = 66

Grading

Advanced

- Each advanced assignment is worth *at least* 10 points
- That means you only need to do 4 to get all 40 points
- You can also do *one* project for a total of 40 points
  - You can submit *one* partially completed project for partial credit
- **Similarly, points past 40 are worth 50%: an 11th assignment would only be worth 5 points**
- If you do 12 advanced assignments and the project...
  - 12 * 10 + 40 = 160 -> 40 + 120/2 = 100: no need to do basic assignments 😃

Grading

tl;dr you get points for each assignment and your letter grade is based on the total points

Strategies for success

- Grading scheme is very flexible
- It's on you to keep up
    - **Schedule some time to watch the videos and stick to it as if it were a lecture**
    - **Take notes!** It engages you more! Don't fall into the "I can watch it later" trap!
    - Class time will give you time to complete homework
    - Doing the video quizzes will add up in the end, enough to save you from multiple assignments!

Any questions?

Demo

Why are you here? How'd you hear about this class?

What's your relationship with computers up until this point?

What's your goal with EECS?

Addenda

Environment

- Terminal emulator: **Alacritty**
    - Former rxvt-unicode user until I learned how bad its font handling was
- Shell: **Zsh**
- Window manager: **i3-gaps**
- Compositor: **picom**
    - Does window transparency effects
- Notification server: **dunst**
    - Displays notifications

Software

- Editor: **Vim**
- System monitoring: **htop**
- Notetaking: **Xournal++**