CSE 130: Principles of Computer System Design

Fall 2019

Basic Information

Lectures:	TuTh 8:00–9:35 AM (Thimann Lecture 003)
Instructor:	Professor Ethan L. Miller (elm+cse130⟨⟨*at*⟩⟩ucsc·edu)
	Office: 337A Engineering 2
TAs:	Allen Aboytes (aaboytes+cse130⟨⟨*at*⟩⟩ucsc·edu)
	Kenneth Chang (kchang44+cse130⟨⟨*at*⟩⟩ucsc·edu)
	Induri Venkata (vinduri+cse130⟨⟨*at*⟩⟩ucsc·edu)
Prerequisites:	CMPE 12 and either CMPE 13 or CMPS 12B; knowledge of C programming language
Required text:	*Principles of Computer System Design*, Saltzer & Kaashoek (ISBN 0123749573)
Home page:	https://canvas.ucsc.edu/courses/24366
Piazza:	https://piazza.com/ucsc/fall2019/cse13002

Course Overview

Students who successfully complete this course will gain an understanding of basic principles of computer systems design: modularity / abstraction, synchronization / concurrency, naming, and performance evaluation. They will understand multiple approaches to each design principle along with the background necessary to choose the right approach for a given situation. They will also understand how to apply these approaches to sample problems, which they will then be able to generalize to computer systems they encounter later.

Students will also gain experience implementing computer systems code. This experience will be invaluable in later courses such as embedded operating systems (operating systems kernel internals), database design, computer security, and distributed systems.

Course Outline

- Introduction: dealing with complexity in computer systems
- Organization of complex computer systems: fundamental abstractions
- Modularity: clients & services
- Modularity: virtualization (memory, CPU, computer)
- Synchronization & concurrency: definitions & primitives
- Synchronization & concurrency: usage examples
- Approaches to naming in computer systems
- Measuring and evaluating computer system performance
- Security in computer systems: issues & mitigation

Where possible and appropriate, we will use examples from a wide range of modern complex computer systems to illustrate concepts covered in class.

Prerequisites

The formal prerequisites for this class are **CMPE 12** and *either* **CMPE 13** or **CMPS 12B**, as well as familiarity with programming in C, which is taught in both CMPE 13 and CMPS 12B. Students are also expected to be familiar with tools such as `make` and `git`, but we will hold tutorials for those who would like a refresher.

While we expect you to know the material covered in the prerequisite classes, we realize that you may not remember all of it. This is OK, but *you* are responsible for reviewing the material—we won't have time to cover prerequisite material in lecture or section after the first week.

Resources

We will use the textbook *Principles of Computer System Design*. We expect you to read the assigned sections in the book *before* the lecture in which we cover the material, so you're prepared to ask questions and discuss the material.

Other than the textbook, all material for the class will be distributed and managed online using Canvas or Piazza. Canvas is used for assignments, grading, and distribution of "official" material, while Piazza is used for discussion and announcements.

Assignments and Exams

Exams

There will be an in-class midterm on November 5th, and a final exam during the scheduled slot during exam week. **You must take each exam at the scheduled time unless you are ill or have an *unexpected* family emergency**. ***You must let the professor know by email or text message before the exam's scheduled start*** regardless of the reason, and **you must provide a doctor's note or letter from the funeral home** before you can make up the exam. There are no exceptions to this policy.

Programming Assignments

Programming assignments are an important component of this course, since they will give you an opportunity to "learn by doing". Your assignments *must* be done on the Ubuntu LINUX 18.04 operating system—that's where we're going to grade them. Since few students run Linux natively, we expect that you'll install a virtual machine on your personal computer to run Ubuntu LINUX 18.04. We'll cover this in section the first week of class.

Programming assignments will all be done individually, and are expected to adhere to the coding standards document available on Canvas.

Rather than approve extensions on a case-by-case basis, we're giving each student 3 "grace days" that may be used, no explanation necessary, to extend the due date of an assignment by 24 hours (this includes weekends and holidays). You may use any number of remaining grace days on an assignment—you don't have to use them all at once. Once you're out of grace days, late projects will lose 25 points (out of 100) per day, with a minimum score of 5 points. Assignments must be submitted within 72 hours of the original due date, regardless of grace days used. **All assignments must be submitted by 8 PM on Saturday, December 7th; assignments submitted after this time will not be counted towards your course grade.**

You must turn in each assignment to pass the class. Assignment submission is done via a `git` repository on GitLab@UCSC and commit ID that passes the minimum standards (indicated by a green checkmark on the GitLab@UCSC GUI). In particular, your code files must successfully compile, even if your program does nothing more than print "Hello World!", and you must have a reasonable attempt at a design document. (Note that printing "Hello World!" won't get you many points towards correct functionality....)

IMPORTANT: If a commit ID you submit for grading doesn't meet the minimum requirements, as indicated by a green checkmark on the GitLab@UCSC GUI, you will get a maximum score of 5 points on the assignment. **VERIFY THE CHECKMARK BEFORE SUBMITTING YOUR COMMIT ID FOR GRADING!!!!**

To encourage you to start early and turn your material in early, assignments will get a 1 point *bonus* per (full) 8 hours they're submitted early, up to a maximum bonus of 9 points. So, if you turn in your assignment 20 hours before it's due, you get 2 points added to your grade. The bonus can't raise your base grade (before extra credit) above 95, and is unavailable if you're using grace days for the assignment.

Written Homework

There will be a few (required) written homeworks during the quarter. They will be graded, and will count towards your final course grade.

Notes & Class Participation

You may, optionally, submit written notes that you take on the material covered over the quarter. These notes may be handwritten or typed *by you*—**you may not turn in material copied verbatim from other sources, including class slides, textbooks, other students, or the Internet**. Notes may be submitted on Canvas by 8 PM on Sunday evening following the week in which

the material was covered in class. We'll use the higher of your notes grade and your exam average for 7% (out of 50%) of your overall exam grade. Submitting notes can only help your grade—you're not penalized if you don't turn them in.

We'll give extra credit for actively participating in class and section, and on Piazza.

Grading

Grades in the class will be assigned as follows:

- Assignments (programming & written): 50% (not weighted equally by assignment)
- Midterm: 16%
- Final: 34%

The *necessary* requirements for receiving a C or better are:

- Get at least a 55% weighted average on your exams. A low grade on one exam can be countered by a good grade on the other exam.
- Get at least a 55% average on your assignments (programming and written).
- Submit a reasonable attempt at all of the programming assignments, including a design document and a `git` repo that passes the minimum requirements check.

Note that meeting all of the necessary requirements isn't *sufficient* to pass—a 57% on exams and 56% on assignments will almost certainly result in a failing grade in the class.

We expect to use the following approximate ranges for overall scores. Individual assignments may be curved, but there is no guarantee of this.

- A: 89–100%
- B: 79–89%
- C: 69–79%
- D: 60–69%
- F: below 60%

Accommodations for Students with Disabilities

If you qualify for classroom accommodations because of a disability, please get an Accommodation Authorization from the Disability Resource Center (DRC) and discuss it with the professor *in person* outside of class (*e.g.*, office hours) **within the first three weeks of the quarter**. We might not be able to accommodate DRC requests made after the first three weeks of class, and DRC accommodations may not be applied retroactively. Please note that **DRC accommodation requests sent directly to the department will only permit extra time on exams**. *All other requests must be discussed with Professor Miller **in advance***. We make every effort to accommodate the diverse needs of our students, but we cannot guarantee that we can accommodate *every* need without first discussing it in person. For more information on the requirements and/or process, you are encouraged to contact the DRC at 459-2089 (voice), 459-4806 (TTY), or at `https://drc.ucsc.edu/`.

Attendance

We won't usually take attendance in class, except for the first week. However, **attendance is mandatory**, and you'll find it difficult to take good notes if you're not in class. Lab section attendance is not required, though you'll miss important material on the programming projects if you don't attend one section per week, since that's where we'll discuss projects in detail. You need not attend the section for which you registered.

Students who wish to have laptops open during lecture must sit on the sides or back of the classroom to avoid distracting other students. **You may not take video, still photos, or audio recordings without written or email permission from the instructor *beforehand*.**

Lecture videos for the week (slides and audio) will be posted on the Monday following the week in which the lectures took place. Please use these videos to augment any notes you took during lecture.

Getting Help

The material in this class can be complex and difficult, so there are several ways to get help with concepts covered in class, homework, and programming projects, listed in approximately the order you should try them for help.

- Attend classes and lab sections.
- Read (really read!) the assignments and other course materials.
- Read and post to the class discussion forum on Piazza
- Meet with the course staff during office hours.
- Email the course staff (`cse130-staff`⟨⟨*at*⟩⟩`ucsc·edu`).

You're encouraged to post **general** questions to the Piazza forum, and to answer questions others have posted there. Asking things like "how does this concept work?" or "can someone help install Ubuntu LINUX 18.04 on VirtualBox?" are fine. Questions such as "can someone post sample code for Assignment 2" or "why doesn't the attached code work?" are ***not*** acceptable, and should be asked during office hours (preferable), or via email. Course staff will also read the forum and reply to posted questions.

Office hours (listed on the Canvas page) are your chance to ask the course staff in-depth questions about the material being covered, programming assignments (including debugging help), or anything else about computer system design or other general computer science issues you want to discuss. Many students find that discussions in office hours are highly informative and interesting, and it usually helps faculty members write you better recommendations for jobs and graduate school. Plus, **I've got an espresso machine in my office, and am happy to give you a shot of espresso to drink while we're talking**. However, please don't just drop by outside of office hours, since I may be busy. If you can't attend office hours, arrange a meeting *in advance* by emailing the person with whom you want to meet.

Email to the course staff will be answered if possible, especially if it only requires a short answer. Questions like "why doesn't my code work?" and "please explain this concept to me" can't be answered via email, so you'll get a brief "come to office hours" response. Email will typically get a response by the end of the next business day.

Academic Integrity

Academic integrity is a requirement for this course (and, indeed, for your entire academic career). All material submitted for a grade must be your own independent work; this includes quizzes, notes, homework, and exams. If you get help on an assignment from anyone other than course staff (professors, TAs, course tutors), you *must* acknowledge their contribution on your submitted work. This *includes* help from tutors from MSI and similar programs as well as private tutors—they are not considered part of the course staff. Obviously, any form of collaboration *during* a quiz or exam is strictly forbidden, but studying in groups for exams is encouraged.

You're encouraged to discuss class material and concepts with others, and you may discuss *general* approaches to programming assignments. **These discussions may not include specific details**, but instead should focus on *how* to approach the project. **You *may not* look at or discuss someone else's design document or code, and you *may not* show your design document or code to anyone else (other than the course staff).** All collaboration must be whiteboard or discussion *only*. You may take handwritten notes from these discussions, but you **may not mechanically record your discussion**—no typing on a keyboard, photos, video, or audio recording during the discussion. (Students with DRC accommodations should discuss note-taking with me *at the start of the quarter*.)

By taking this class, you agree to abide by the Personal Responsibility Document that's available on the course Canvas page. **You must submit a signed copy of this document on Canvas, by October 4 (Friday) at 5 PM.** You must turn in the signed Personal Responsibility Document before submitting any assignment to be graded.

We take academic integrity very seriously, and report *all* violations of academic integrity to the School of Engineering and to your college Provost. If you violate academic integrity, you will fail the class. Period. Depending on the severity of the violation, the university may impose additional penalties, including suspension and even expulsion in rare cases.

The bottom line: **don't cheat!**

