Systems II: Operating Systems
Spring 2026
Course Information

Prof. Kaplan

Last updated: 2026-Jan-25

Be sure to read **all** of this document!

1 The topics

Computing hardware—the processing cores, memory, storage, networking, input devices and video displays—are marvelously powerful, efficient, inexpensive devices. In their raw form, however, they are remarkably difficult devices for a person to **use**. Each is controlled by highly detailed arrangements of binary signals that reveal more about the inner engineering and capabilities than they reflect any human-driven use to solve computational problems.

Enter the *operating system*: a base layer of software that acts as an intermediary between the complex hardware interfaces and the programmers seeking to use that hardware to perform actual, useful computations. We will examine the fundamental structure and function of a traditional operating system kernel, which is the heart of this intermediate layer. We will rely on an *instruction set architecture* (as developed in Systems-I), which provides the interface between the hardware layer (the processor) and this lowest-level of software. Our goal will be to make a general-purpose system on which someone can solve algorithmic problems via programming without needed to manage the hardware itself.

Below is a brief list of the core topics that we will cover, in **roughly** the order that we will cover them. For most of these, we will examine them in detail by implementing them.

- Instruction Set Architectures (ISAs), assembly and machine code
- Booting
- Processes and CPU scheduling
- Virtual memory mapping and swapping
- Concurrency and synchronization support
- File systems, block devices, and persistence
- Virtual machines and hypervisors

This course will be project-intensive. We will not merely talk about these concepts; implementation will be a central experience as we build a simple, working (simulated) RISC-V OS kernel and supporting tools. These projects will require us to solve a number of algorithmic problems, and to develop working components of the lowest levels of a system.

This course should be fun, as there is a great deal of hands-on experience with the material. It is also a great demystifying course, as you will have a much better understanding of the operation and principles behind the computers that surround us. Note, however, that it is a course with a great many details, as well as a course that is exceedingly cumulative. It will be critical that you stay on top of lectures and labs at all times.

2 Lectures and labs

This class meets **MW** from **11:35 am to 12:50 pm** in **SCCE A331**. Our time will be a mixture of lectures/discussions and lab time for the projects.

You are expected to be present for **all of the lectures and labs**; missing either is strongly discouraged. I will not teach material twice, so if you miss a lecture or a lab, you're on your own. If you must miss lecture or lab due to an illness, a curricular conflict (e.g., a Geology field trip), or for some emergency, then contact me and we will arrange to handle the situation. **If you have a conflict** with a lecture or lab—for an athletic event, performance, or other extra-curricular activity, or to depart early for or arrive late from a vacation—then **the choice is yours to miss or to attend**. If you choose to miss the class meeting, I do **not** want to know *why* nor even *that* you are missing class. You have elected, voluntarily, not to attend, and you must be prepared to obtain and learn the material that you missed on your own. I, of course, recommend that you choose to attend the class meeting when these conflicts arise. Do not underestimate the willingness of those who run extra-curricular programs to make accommodations for your academic priorities.

I expect you not only to attend lectures and labs, but also to be attentive for them. The time will be best spent if it is interactive, and that requires that you be up-to-date on the class material, and that you be alert and prepared to participate.

3 Texts and materials

The text for this course is available through the college's textbook program; it is also freely available online:

Operating Systems: Three Easy Pieces by Remzi and Andrea Arpaci-Dusseau

Any other essential material will be posted to the course web pages and Slack channel. Additionally, you are encouraged to use any good material that you find for yourself online. You cannot, of course, plagiarize code—and using an AI system to generate code is plagiarizing—but you should take advantage of the many books, lecture slides, problem sets, etc., that can help you to more deeply understand each topic.

4 Assignments, deadlines, and extensions

There will be a number of projects. The deadline for each will be stated clearly on the assignment. Project work will be more-or-less continuous throughout the semester, so falling behind and catching up is not a practicable option. When a deadline arrives, **turn in what you have**, whatever its state, so that you can move onto the next project.

An extension for any assignment **must be requested, in writing** (email counts as *writing*), **at least 48 hours prior to the deadline**. The determination as to whether or not a particular situation merits an extension will be made on a case-by-case basis. Scheduled events are **not** sufficient reason to warrant an extension. Rather, extensions are intended for unusual circumstances that prevent you from planning your time well in order to meet the deadline. Note that a sudden onset of illness or other emergency situation that occurs less than 48 hours before a deadline will be treated as a special case.

5 Grading

In addition to the many projects, there will be a scheduled final exam.

Your final grade will be chosen by my evaluation of how well you have mastered the course material at the semester's end. All of the work that you submit, your final exam, and your participation in class all contribute to my impression of that mastery.

6 Academic dishonesty

You will be expected to do your own work on all assignments and exams in this course except where explicitly noted on group assignments. While I encourage you to interact with your classmates and discuss the material and assignments, there is a limit to the specificity of such discussions. I seek to make that limit clear here.

It is acceptable to discuss any assignment for the class with a classmate. You may even discuss your approach to a particular problem, or review relevant material for a problem with another person. However, you **may not show another student your work, nor see another student's work. If in doubt, *ask me***. If you are unsure whether or not a particular kind of communication would rise to the level of academic dishonesty, then you should contact me immediately and find out.

Additionally, any code that you submit must be written by you. Submitting code produced by someone or something else (whether from another student, from a web site, or from an AI system) is plagiarism.

Regarding the use of AI's or other internet sources: Don't. The core experience of this course is to internalize the concepts, logic, and structures involved in designing and implementing an operating system. Engage fully with that project and you will understand much more clearly and deeply. Embrace the struggle to build a clear understanding of the concepts and their implementation. Share that struggle with your fellow students as they do the same. Don't ask an AI, because it was already trained on all of this background and doesn't need to learn it (and doesn't know how to struggle). Using AI steals from yourself the work that changes your brain. Don't do that. Change your brain.

7 The big picture

There will be so many details to remember and understand in this course that it is easy to miss the forest for the trees. Don't forget, now and then, to consider the bigger picture: that from one simple level of capabilities, you can create another, fundamentally different, more complex level. As you repeat this process, you develop something so complex from components that are so simple that it seems nearly impossible that the former could arise from the latter. Only because we have seen the progression, step by step, do we see how it is done.