

CSC213 Operating Systems and Parallel Algorithms

(Fall 2008)

Synopsis: This course is a bridge between the past and the future. We will learn to appreciate abstractions already provided by operating systems, which will help you become not only proficient users of modern systems (from a programming perspective), but also effective designers of your own abstractions. As multiprocessor systems become more prevalent and more important, the principles of synchronization and parallelization will become essential knowledge for building systems that can effectively harness the ongoing developments in computing power.

Instructor: Jerod Weinman
Office: Noyce 3825
Phone: x9812
E-mail: [weinman]

Office hours:

Monday	1:00-2:00 PM
Tuesday	3:30-4:30 PM
Wednesday	4:00-5:00 PM
Friday	2:00-3:00 PM

or by appointment.

Course web page: <http://www.cs.grinnell.edu/~weinman/courses/CSC213/2008F/>

Overview

“Operating Systems” should really be called “Software Systems.” This course will help you understand the abstraction from a hardware-based machine to the software-based world of the programmer. Computational complexity is one (very important) aspect of analysis, but another critical aspect is “the system.” Learning about what is abstracted can help you debug your program’s *performance* on a system.

Programs large and small will use an OS. Many growing application areas (e.g., e-commerce, social networking, information management/retrieval) and a diverse array of fields (such as biology, climatology, physics, and ecology) incorporating more computation will require large systems that effectively employ multiple CPUs, disks, network interfaces, etc. Expanding the computational frontier will require fluency in the areas of synchronization, parallelization, and distributed algorithms.

Perhaps it is you who will soon help craft a new language that makes writing parallel programs to exploit multiple CPUs as easy as it is to write a file to magnetic disk today (as opposed to the punch cards, assembly code, and device driver bit-twiddling of yore.)

Our major objectives for this course include:

- Learning to identify, understand, and *practice* the principles of abstraction behind OS design that are supplied to the programmer

- Understanding modern OS design principles that enable you to build effective large-scale, networked, or multi-threaded applications
- Fluency in the elements synchronization and concurrency along with design principles for parallel processing

Why take it? If you have any interest in advanced computer science study, high-performance computing, or adding a twist of computation to just about any other science, this course will be highly relevant and rewarding for you! This course fulfills the CS major's systems requirement.

What do I need to know? This course assumes you have taken CSC 201. That means you should be reasonably adept at

1. the practice of programming (problem analysis, program design, debugging, etc.).
2. programming in the C language

If you've taken any of these courses you will be equipped to thrive in this course. If you're a little rusty on the C, you should review your earlier class material, Nutt's background material on the student website (<http://www.aw.com/nutt>), or Walker's *An Introduction to C Through Annotated Examples* (<http://www.cs.grinnell.edu/~walker/c>), and keep a C reference handy. (After all, even writers who are perfectly fluent speakers of English keep a dictionary or style manual close by when practicing their craft!)

Textbook

Our main textbook is:

Gary Nutt. *Operating Systems*, Third Edition. Addison-Wesley, 2000. ISBN: 0-201-77344-9

You may also wish to acquire a good C language reference and keep it handy. One trusty example is:

Samuel P. Harbison and Guy L. Steel. *C: A Reference Manual*, Fifth Edition. Prentice Hall, 2002. ISBN: 0-13-08952-X

Occasionally our discussions will be supplemented by other texts or research papers. I will provide these and note them in the detailed class schedule.

Class Meeting

Class	MWF	8:00-8:50 am	Science 3819	
Lab	T	9:00-9:50 am	Science 3819	L1
		2:15-3:05 pm	Science 3819	L2

Short version: You are expected to attend and actively participate in class. I am expected to make class worth attending.

Longer version: To make class time most valuable for you, I do not plan to lecture on material that is covered in the textbook. Therefore, you should come to class prepared to clarify, discuss, and practice the ideas in the text by reading any assigned material before class. To facilitate this, you will be contributing your own questions on the readings for discussion throughout the term. In this way, every class can be tailored to the areas you are most curious or confused about.

Reading Suggestions

Before each class, you should check the class schedule for updates and do any reading that has been assigned. Reading the assignment entails the following.

Overview You should skim through the reading once to get an overview of the material to be covered. The first reading can (and should be) very quick. (*Expected time: 10 to 15 minutes.*)

In-Depth Next, read the material closely. Not everything will make sense at this point, but hopefully many things will. (*Expected time: 30 to 45 minutes.*)

Final Notes After carefully reading the material, try making a few notes to yourself about what you think are the most important concepts being covered, as well as any questions you have. (*Expected time: 5 to 10 minutes.*)

Attendance Policy

Our class meets early, and I know that sometimes “things happen.” Therefore, you will be granted **one** absence from class (not labs) with no penalty for participation. While I do not take attendance, I will tend notice your presence or absence as a trend, and this can make a large contribution to the participation portion of your grade. If you wish me to acknowledge your absence as excused, you must either:

1. Notify me at least 7 days in advance to make arrangements for your absence, *or*
2. Provide me with documentation of your absence’s circumstances *post hoc* from Health Services or Student Affairs.

It is very important for you to attend labs because you will work in teams. Don’t let your partner (or yourself) down!

If you do miss a class, you must first talk to a classmate about any material that you may have missed. After that, you may follow up with me about any further questions or concerns.

Schedule of Topics

The following is an approximate schedule of topics to be discussed during the course. See the web page schedule for details.

Week	Topic
1	OS overview
2	Computer organization
3	Processes and threads
4	Process scheduling
5	Synchronization
6	Deadlock
7	Memory management

Week	Topic
8	Networks and sockets
9	Parallel computation
10	Parallel analysis
11	File management
12	Protection and security
13	Distributed computation
14	Current OS topics

Assignments and Activities

Under a normal 16 credit load, I expect that you will spend at least 40 hours per week on your studies (class time, homework, and studying). Thus, you should plan to spend 10 hours/week on work for this course. With class and lab time clocking in at $3\frac{1}{3}$ hours, you'll have $6\frac{2}{3}$ hours/week left for the following:

Discussion Questions Throughout the term, you will be expected to submit an average of one discussion question per class meeting that pertains to that day's reading. See the course web page for guidelines on question content and submission.

Brief Exercises Questions that engage you in design justification and/or a deeper understanding of abstractions will be assigned regularly at the end of class and posted on the web page. (No more than one question per class—thus, 1-3 per week). These are due by the *beginning* of the following class (not the end, not even ten minutes in). If you have a clear understanding of the core issues, your answers should not be longer than one page (they will frequently be shorter, depending on your handwriting or word processor formatting), and they should not take you very long (perhaps 30 minutes) if you have a solid understanding of the issues. It is also my intent that these questions will be similar in nature to many that will be on the exams.

Discussion of the class material with your classmates is of course encouraged. However, collaboration is not permitted on the exercises; engaging with the material yourself to discover what the relevant issues are will greatly increase your learning. The idea is for you to frequently practice analyzing system considerations and get feedback in a low-stakes framework.

Grading of the exercises will be on a simple ternary scale:

- ✓ for an adequate, sufficient response
- ✓+ for a particularly clear or insightful response
- ✓- for an unclear or insufficient response

I expect most work will receive checks. Of course, no credit will be given if no response is submitted. No late submissions will be accepted.

Significant Bits From time to time (typically on Mondays), one member of the class will present a **five minute** overview/preview/insight/review of a recent development in the broad area of "systems." The idea is to establish a practice of following technological developments in the popular and technical press as well as research publication venues. In doing this, you will inform your classmates, at a high level, about something that relates to the course material in principle and goal, rather than necessarily a particular topic. Your presentation should include **what** the development is, **why** it is important, a bit about **how** it is/was done, and perhaps something on **who** did it. You will be asked to do just one

during the semester, so find something interesting and share it with us! See the course web page for the schedule, details, evaluation rubric, and suggestions for finding topics.

Laboratory Work We will practice using programmer-level OS elements via the UNIX system call interface and C standard library in our weekly lab sessions. Since lab time is limited, it is very important for you to **come prepared** by reading the assignment and any other related material beforehand. You will start your work in our weekly sessions but complete it outside of lab. Assignments may be broken into parts with multiple due dates to help you structure your time. You will typically complete these labs in pairs or groups, oftentimes assigned by me on a rotating basis, and only one lab will be submitted for the group. Everyone whose name appears on a submitted lab report has the responsibility to ensure everyone fully understands the submission.

Lab reports will typically be due the following Friday and/or Monday after the Tuesday meeting. See the course webpage and the particular lab for submission guidelines and formats.

You are not limited to collaborating only with the members of your group. Discussion of concepts and approaches with other classmates is encouraged. Debugging C programs and concurrent algorithms can be difficult and is often helped by explaining your code to someone else (which will also frequently help you to explain the bug to *yourself*). All such contributions by others (not in your group) should be properly attributed in your report. Furthermore, all the work you submit (code, experimental data, write-ups, etc.) must be that of the group. Code provided by the instructor or the text should be attributed, but no other code or written work (from any source) may be shared with others or copied for your own use.

Exams As opportunities for you to demonstrate your design prowess and grasp of OS abstraction principles, there will be two hour exams as well as a cumulative final exam.

Hour Exam 1	Friday, October 3
Hour Exam 2	Friday, November 14
Final Exam	Tuesday, December 16 (2 PM)

Do not make airline reservations that will conflict with your final exam schedule.

You are highly encouraged to use the PioneerWeb Discussion Board for questions related to the course or laboratory exercises. If a post is related to an assignment, it must adhere to the standards of collaboration for that particular assignment. If you are debugging, one good rule of thumb is to not post code; however, it is possible that error messages may be enlightening to others. (I often find that when I am explaining some error by writing up a post or email, I typically find its source.)

Grading

My goal is for everyone taking this course to be able to demonstrate familiarity and fluency with the course concepts. I would be very happy if you all met the goals above and received “A”s. The following weighting will provide a basis for evaluation,

Laboratory Work	45%
Exercises	10%
Discussion & Participation	10%
Hour Exams	20%
Final Exam	15%

with the caveat that you must pass the final exam to pass the course.

Academic Honesty

You, as students, are members of the academic community. Both the College and I expect the highest standards of academic honesty. (See the Grinnell College Student Handbook, e.g., <http://www.grinnell.edu/offices/studentaffairs/shb/section3/academichonesty>). Among other things, this means clearly distinguishing between work that is your own, and work that should be attributed to others. It is expected that the collaboration policies given in this syllabus and on particular assignments will be followed. Furthermore, any program results or output must be faithfully recorded, not forged. (A thoughtful explanation of unexpected behavior can often be a worthwhile submission and is much better than the alternative.)

As an instructor, I will meet my obligation to bring any work suspected to be in violation of the College's Academic Honesty Policy to the attention of the Committee on Academic Standing, after which there is no recourse with me.

Deadlines

Assignments are due at the *beginning* of class (or lab) on the specified date. Exercises and laboratory reports due on days for which you have a prior excused absence must still be submitted by the deadline (via email or classmate proxy).

For the laboratory reports, a late penalty of 33.33% will be deducted at each subsequent class or lab meeting. Thus, you have at most two additional meetings to submit your work.

Because the brief exercises are intended to be relatively frequent, low-stakes assignments, no late submissions will be accepted.

Exception: Deadlines for programming-based assignments will automatically be extended by at least one class period if MathLAN is down for an unscheduled period of 3 or more hours during the week preceding the assignment due date.

Contacting Me

Please come by during my office hours to discuss the course content, get any extra assistance, or just talk about how the course is going. Note that if multiple students have similar questions or issues, we may work together as a group. If you cannot attend a scheduled office hour, you may also email me to schedule an appointment; please include 3-4 possible meeting times so that I can pick one that works for me.

I enjoy getting to know my students, but I prefer to reserve office hours for academic matters. If you would like to have a more informal conversation, I would be delighted to accept an invitation to lunch (<http://www.grinnell.edu/offices/studentaffairs/chaplain/religiousandspiritualprograms>).

Email is also a reliable way to contact me, but please allow 24 hours for a response (except on weekends, when I do not regularly read email). You may also call me in my office (x9812) for more urgent matters (e.g., you will be missing a lab due to illness).

Accommodations

If you have any disability that requires accommodations, please meet with me right away so that we can work together to find accommodations that meet your learning needs. You will also need to provide documentation of your disability to the Dean for Student Academic Support and Advising, Joyce Stern, located on the 3rd floor of the Rosenfield Center (x3702).

With thanks to Janet Davis for the “Reading Suggestions” and other key policies.