

CMPS 111: Introduction to Operating Systems

Spring 2009

Basic Information

Lectures: Tuesday & Thursday 12:00–1:45 PM in 194 Engineering 2
Labs: Tue 2:00–4:00; Wed 11:00–1:00 (both in Baskin 109)
Instructor: Professor Ethan Miller ([elm \(a\) cs.ucsc.edu](mailto:elm@cs.ucsc.edu))
Office: 365 Engineering 2
Hours: Wed 4–5 PM and Thu 2–3 PM
TA: Noah Watkins ([jayhawk \(a\) cs.ucsc.edu](mailto:jayhawk@cs.ucsc.edu))
Office: E2-380
Hours: Mon 11–noon and Fri 10–11 AM
Staff email: [cmps111-staff \(a\) cs.ucsc.edu](mailto:cmps111-staff@cs.ucsc.edu)
Prerequisites: CMPS 101 and either CMPE 110 or CMPE 112
Required text: *Modern Operating Systems* (3rd edition), Tanenbaum
Optional texts: *Operating Systems: Design & Implementation* (3rd edition), Tanenbaum
The Design and Implementation of the FreeBSD Operating System, McKusick, *et al.*
Home page: <http://www.soe.ucsc.edu/classes/cmps111/Spring09/>

Course Overview

The goal for students in this course is to learn the fundamental principles of operating systems. To help you accomplish this, we will discuss the various important aspects of operating systems in general, examine specific examples from current operating systems, and do programming assignments with a generic teaching-oriented operating system (MINIX 3). The specific topics we will cover include:

- Basic operating system concepts
- Processes and scheduling
- Deadlocks
- Memory management
- Operating system management of input/output devices
- File systems
- Protection and security
- Introduction to distributed operating systems (if time permits)

Where possible and appropriate, we will use examples from Linux, FreeBSD, and other modern operating systems to illustrate concepts covered in class.

Prerequisites

The formal prerequisites for this class are **CMPS 101** and *either* **CMPE 110** or **CMPE 112**. Students should have taken these classes or equivalent relatively recently; experience has shown that students who took these classes more than two years ago tend to have more difficulty with the material in CMPS 111. You should also be familiar with C programming in UNIX (including the UCSC UNIX systems) using such tools as `gcc` and `makefiles`. You're welcome to ask for help with these tools, but such questions will have lower priority than those about material covered in the course.

Textbook

The required text, *Modern Operating Systems, 3rd Edition*, is available at the UCSC bookstore and at online book-sellers such as Amazon. It really is a required text—homework problems may be taken from the book, and lectures and notes will complement the material as presented there. ***Make sure you get the third edition of the text.*** The optional texts are just that—optional. One covers a popular operating system—FreeBSD—in depth and may

be of interest to those who want more details about operating systems in the real world. Neither homeworks nor projects will depend on the contents of the FreeBSD text, except if the material is covered in class, in which case it'll be in the slides. The other text provides some additional detail on the structure of the MINIX 3 operating system. This material will be very helpful in doing your programming assignments. However, it's not required because some people may not need the extra documentation that the book provides. The book is available online (for a fee) at <http://www.coursesmart.com/>.

Online Resources

Most of the material in this class will be available online at <http://moodle.soe.ucsc.edu/>. You **must** set up an account on Moodle by **Friday, April 3rd**. You'll need your `ucsc.edu` email account to create a Moodle account, but the account is otherwise free. We strongly recommend that you set up an account now even if you're not sure you're taking the class, since you can always "turn off" the account if you don't need it.

Assignments, Exams & Grading

Exams

There will be an in-class midterm in early May and a final during final exam week in the slot allocated by the registrar's office (Monday, June 8th from 4:00–7:00 PM).

Homework

There will be a homework assignment assigned about every week and a half, due about one week later. The homework will give you a chance to see how well you understand the concepts we've covered in class.

Programming Projects

Programming projects are an important component of this course. You'll be making changes to the MINIX 3 operating system, which runs on x86 processors (or simulations of them). We recommend using a virtual machine system such as VMWare or VirtualBox, both of which are available for x86-based systems running MacOS X, Windows, and Linux. In addition, VMWare should be installed on the computers in Baskin 109. The programming tools you'll need (C compiler, debugger, etc.) are part of the MINIX 3 install CD, which, along with more details on the projects, is available from the course Moodle pages. Project descriptions will be posted on the web, and will be accessible from anywhere on the Internet.

Projects 2–4 may be done in teams of two students, both of whom must be enrolled in the class—your friend who works at Microsoft or Google isn't eligible, unless he or she is in the class this quarter. Project partners receive the same grade for the project (modulo days, as described below).

Programming assignments must be turned in on time. Rather than approve extensions on a case-by-case basis, we are giving each student 4 "grace days" that you may use, no explanation necessary, to extend the due date of an assignment by 24 hours (this includes weekends and holidays). Grace days need not all be used on the same assignment—you can use all four on a single assignment, or use one each for four different assignments (or any other combination). Once you've used them up, though, ***late projects will receive a grade of zero***. Period. Use your grace days wisely.

Please make a note in a README file that you're using grace days when you submit the assignment; you can't retroactively apply them to assignments you've already submitted. If you're part of a project team that wants to use grace days, **each person** on the team must use his/her own grace days to extend the due date. For example, if Alice and Bob want a one day extension for the due date for their project, Alice must use one of her days and Bob must use one of his; it's not acceptable to use two of Alice's days. If only one partner uses grace days, the other partners will receive a zero accordingly. Keep in mind that a project team may only turn in a project once; you can't turn in a version on time for one person and two days late for another person on the same team.

You must turn in a reasonable attempt at each project to pass the class. Graded assignments will be returned as soon as possible, usually within one week, and grades will be available online during the quarter.

Class Participation

Your class participation grade is based on several factors: actually participating in lecture (we'll be using iClickers this quarter), visiting office hours, and participating in lab sections. It's only 5% of your grade, so it probably won't determine whether you pass or fail, but it's how we decide whether to give you an A- or B+ if you're right on the border.

We'll be using iClickers this quarter (available at the bookstore) during class to encourage more student participation. When you use an iClicker in response to a question, you'll receive credit towards class participation both for simply answering the question and for answering it correctly (when there's more than one correct answer, *any* correct answer receives credit). You may use any iClicker you want (borrowing one from a friend *not taking CMPS 111* is fine), but you must use the same iClicker every class, and you may not operate more than one—clicking for your friend who's ditching class is considered **cheating**).

Grading

Grades in the class will be assigned as follows:

- Programming assignments: 45%
- Midterm: 17%
- Final: 25%
- Homework: 8%
- Class participation: 5%

To pass the class, you must do the following:

- Have at least a 50% average on your exams. A low grade on one exam can be countered by a good grade on the other exam.
- Have at least a 50% average on your programming projects.
- Turn in all of the programming projects. If you miss an assignment due date and have no grace days left, you still have to turn in a reasonable attempt at the assignment, though you will receive a zero for it.

Note that a 50% average on both exams and projects is not *sufficient* to pass—a 51% on exams and 53% on projects will likely result in a failing grade.

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 submit it to the professor or TA in person outside of class (*e.g.*, office hours) **within the first two weeks of the quarter**. For more information on the requirements and/or process, contact DRC at 459-2089 (voice), 459-4806 (TTY), or at <http://drc.ucsc.edu/>.

Attendance

Class attendance is mandatory; we typically won't take attendance, but class participation *is* 5% of your grade. Homeworks, projects, and important dates will be posted to the Moodle site, but this is provided as a courtesy and may lag class by a day or two. *It's your responsibility to find out what you missed if you don't attend 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. This is where the programming projects will be discussed in detail. Office

hours are optional. They're your chance to ask the professor and TA questions about the material being covered, programming assignments, or anything else about operating systems (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.

Getting Help

Operating systems is a tough subject, 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
- Check the class Web page frequently (every day or two)
- Read and post to the class discussion forum, available off the Moodle site
- Meet with the professor and TA during office hours
- Email the professor and TA (`cmeps111-staff (a) cs.ucsc.edu`)

Please meet with the professor or TA in person if you have an in-depth question, such as detailed help with the programming assignment design or debugging. Please don't just drop by outside of office hours; if you can't attend office hours, arrange a meeting *in advance* by emailing the person you want to meet with.

You're encouraged to post **general** questions to the discussion forum. Asking things like "how does this concept work?" or "can someone help install MINIX 3 on VMWare" are fine. Questions such as "can someone post sample code for Project 2" are *not* acceptable. Please ask such questions during office hours (preferable) or via email.

Email to the course staff will be answered if possible. The best kinds of questions to ask via email are those that require short answers. Questions like "why doesn't my program work?" and "please explain this concept to me" are much more difficult to answer via email, and are best asked and answered in person at office hours. Please keep in mind, too, that email replies may take several hours, depending on when the course staff read and respond to email.

Office hours are listed at the start of the syllabus. These hours are subject to change; if you would like to see office hours at a different time, please let us know.

Academic Honesty: Collaboration vs. Cheating

In an ideal world, academic honesty wouldn't be an issue. Sadly, we don't live in an ideal world.

You're encouraged to discuss the course material and concepts with other students in the class. If you do so, you may not take notes during the discussion. You should also follow the *Simpsons* rule¹ when studying course material with others. If you get significant help on an assignment from a source other than the textbook, slides, your own notes, or the course staff, you *must* acknowledge it in whatever you turn in. It's not cheating to read other textbooks or look for help on the Internet; it *is* cheating if you copy solutions from one of these sources. Under no circumstances may you look at anyone else's code or show anyone else your code (except for your project partner, of course). While you may discuss the concepts used in the programming assignments, you may not discuss implementation details of the assignments themselves. If you are caught copying or otherwise turning in work that is not solely your own, ***you will fail the course and a letter will be sent to your Department, the School of Engineering, and to your Provost and academic preceptor.***

The bottom line is that you're expected to conduct yourself as a person of integrity—you're expected to adhere to the highest standards of academic integrity. This means that plagiarism in any form is completely unacceptable. You are a (soon-to-be) computing professional; we encourage you to consult the code of ethics appropriate to your discipline (the ACM, IEEE, and Computer Society all have resources on this).

Plagiarism will be assumed until disproved on work that is essentially the same as that of other students. This includes identically incorrect, off-the-wall, and highly unusual duplicate answers where the probability of a sheer coincidence is extremely unlikely. All parties to this unacceptable collaboration will receive the same treatment. In the case of programs, reordering routines or simply renaming variables does *not* make two programs different.

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

¹The *Simpsons* rule states that you should watch an episode of *The Simpsons* between discussing the material with other students and working on your own assignment; other 30 minute activities, such as watching *Futurama* or *Family Guy* are also acceptable.