EECS 482: Introduction to Operating Systems
Winter 2017

1 Basic information

Lecture: MW 1:30-3pm, Stamps Auditorium

Instructor:

Jason Flinn,	jflinn@umich.edu,	office hours MW 9-11am, 4641 BBB

Course staff:

Josh Cheng,	chengsta@umich.edu,
Zihuan Diao,	diaozh@umich.edu,
Manav Gabhawala,	manavg@umich.edu,
Margo Leibold,	margolei@umich.edu,
Maxwell Morrison,	morrimax@umich.edu,
Joel Sharin,	jsharin@umich.edu,
Aman Sharma,	amsharma@umich.edu,
Christian Smith,	cesm@umich.edu,
Owen Yang,	owenyang@umich.edu,

Note: All office hours in 1695 BBB, see class Google calendar for times.

Note: office hours are subject to change, to adapt to the needs of students and the instructors. Check the discussion forum and class calendar for the latest schedule.

E-mail alias for the course staff: eecs482@umich.edu
Course web page (Canvas): http://canvas.umich.edu/
Discussion forum (Piazza): http://piazza.com/umich/winter2017/eecs482

2 Course overview

EECS482 is an introductory course in operating systems at the senior undergraduate or first-year graduate level. The objective of the course is to familiarize you with the issues involved in the design and implementation of modern operating systems. The concepts in the course are not limited to any particular operating system or hardware platform. We will discuss examples drawn from historically significant and modern operating systems including MULTICS, UNIX, Mach, Linux, and Windows. We will cover topics such as processes and threads, concurrency and synchronization, CPU scheduling, virtual memory management, communication in distributed systems, secondary-storage management, file systems, transactions, replication, and fault tolerance.

To help you understand operating systems, you will implement several modules that form much of the core functionality in modern operating systems. These projects will give you practical exposure to topics such as threads, virtual memory management, client-server systems, and file systems. We will also assign homework questions that will be discussed in discussion sections.

3 Prerequisites

Students must have completed EECS 281 and EECS 370. Students are expected to understand computer architecture and data structures, to have extensive C/C++ programming experience, and to be familiar with UNIX.

4 Course material

The recommended text for the course is *Operating Systems Principles and Practice* by Thomas Anderson and Michael Dahlin (ISBN 0985673516). There will also be lecture notes and supplementary readings posted on the course home page. Course announcements and project help will be posted in Piazza.

5 Class projects

Four projects will be assigned during the term, each of which will require a substantial time commitment on your part.

5.1 Group policies

Three of the four projects in this course are group projects (project 1 is done individually). You must form a group of 2-3 students for the last projects 2-4. Members of a group need not be in the same discussion section or lecture. Declare your group's membership at https://grader2.eecs.umich.edu/eecs482/group.php. The group declaration deadline is Friday, January 20th (shortly after Project 1 is assigned). After that date, we will combine remaining students into groups of 2-3. It is in your best interests to choose your partners carefully. You should discuss topics such as prior experience, course background, goals for this course, workload and schedule for this semester, preferred project management, and work style. Make sure you can find several blocks of time during the week to meet to discuss or carry out the project. If you worked on one or more 482 projects in a past semester, make sure you talk to your professor before joining a group.

Students are expected to participate wholly in their group to the benefit of the entire group. All group members should be familiar with all aspects of the project, irrespective of their role on the project. All group members are required to contribute their fair share, and we expect to assign the same project grade to all group members. To help ensure this, group members will evaluate the contributions of other group members after each project. Members who contribute less than their share may receive a lower grade on the project; non-contributing members will receive a zero. In case of disputes regarding contribution, an instructor may examine the commit log and/or interview group members on the project's design and implementation.

Students may be "fired" from a group by a majority vote of the remaining group members. The procedure for this is: (1) documented "gentle warning" of risk of firing in e-mail, with cc to all group members and to eecs482@umich.edu, with cause and specific work required to remain in the group, (2) allow at least 72 hours for compliance, and (3) send documented statement of firing in e-mail, with cc to all group members and eecs482@umich.edu. Fired group members must actively pursue and obtain membership in another group. Those who do not find partners will need to work alone; it is not in your interest to get fired by your group.

Managing group dynamics and using each group member's time and talents effectively can be as difficult as solving the project. We are happy to offer advice on how to handle these issues. Be open and candid with your group about any personal problems early on so that your group can plan around such problems and not fall behind. A sure way to make your group upset at you is not finishing your part of the work at an agreed-upon deadline *and* not informing them about the problems early enough for them to help. We encourage everyone to read the note "Hints: Couch potatoes and hitchhikers" (posted on the CTools resources page).

All projects will be hosted at github. Please sign up for a free github account as soon as possible, then register your github username with us. The eecs482 organization at github will provide a private repository for each group for each project. Commits to the repository should reflect the proportion of work performed by each group member. If you pair program, take turns at the keyboard so that commit logs reflect the effort of all participants.

5.2 Turning in projects

You will submit your projects through a web form. Projects are due at 11:59 pm on the due date. You should allow for short-term, unexpected events like network outages, computer crashes, submission problems, and clock skew, and not plan to submit your project exactly at 11:59 pm. We will not grant extensions in the above cases.

Sometimes unexpected events make it difficult to submit a project on time. For this reason, each student will have a total of 3 late days to be used for projects throughout the semester. These late days should only be used to deal with unexpected problems such as illness. They should not be used simply to start later on a project or because you are having difficulty completing the project. Once late days are used up, submissions received after the due date will not count (even if they are just one second late). Pro tip: try to save some late days for the last project. Weekend days are counted in the same way as weekdays (e.g., if the project deadline is Friday and you turn it in Sunday, that's two days late). If a student has used up all late days, but other group members have remaining late days, the group

may continue to submit. However, subsequent submissions will only count for those group members with remaining late days.

To request an extension beyond the free late days, you must discuss your situation with an instructor **before** the deadline and provide written documentation. Requests for individual/group extensions are rarely granted, even for computer problems, illness, family emergencies, etc. In most cases, with planning, cooperation and good faith on your part, your group will be able to make up the deficit without needing an extension. Pro tip: you can avoid most problems by starting the projects early and by using version control via GitHub. If a family/personal emergency causes you to miss a significant number of days, please see an instructor to decide the best course of action. If you are having trouble understanding the material or starting a project, please come to office hours for help right away.

Contact your professor at the beginning of the semester if you have a condition that might interfere with your ability to participate in class, submit assignments, or take exams.

5.3 Honor code

All projects in this course are to be done by your own group and in accordance with the College of Engineering Honor Code. Violation of this policy will result in a zero on the project in question. Suspected violations will be reported to the Engineering Honor Council. We use an automated program and manual checks to correlate projects with each other and with solutions from prior semesters.

At the same time, we encourage students to help each other learn the course material. As in most courses, there is a boundary separating these two situations. You may give or receive help on any of the concepts covered in lecture or discussion and on the specifics of C/C++ syntax. You are advised to consult with other students in the current class to help you understand the project specification (i.e., the problem definition). However, you may not collaborate in any way when constructing your solution; the solution to the project must be generated by your group working alone. You are not allowed to work out the programming details of the problems with anyone else or to collaborate to the extent that your programs are identifiably similar. You are not allowed to look at or in any way derive advantage from the existence of project specifications or solutions prepared in prior years (e.g., programs written by former students, solutions provided by instructors, project handouts). If you worked on 482 projects in the past (e.g., if you are repeating the course), you are not allowed to re-use code that you or your group wrote from the prior semester.

If you have any questions as to what constitutes unacceptable collaboration, please talk with an instructor. You are expected to exercise reasonable precaution in protecting your own work. Don't let other students borrow your account or computer, don't leave your program in a publicly accessible directory, and take care when discarding printouts.

5.4 Tips for success on the projects

The most common reason for not doing well on the projects is not starting early enough. You will be given plenty of time to complete each project. However, if you wait until the last minute to start, you may not be able to finish. Start early and plan to have it finished a few days ahead of the due date; many unexpected problems arise during programming, especially in the debugging phase. The computing sites can become quite crowded as deadlines approach, making it difficult to get a computer. Plan for these things to happen. Your decision to start late is not an excuse for turning in your project late, even if unfortunate situations arise (such as having your computer crash).

The most common reason for spending too much time on a project is hacking before thinking. Resist the urge to start banging out code as your first step; it is too easy to code yourself "into a corner". Read the project specification carefully in a quiet place, far away from your computer. Pay particular attention to the project descriptions, and generate a list of behaviors that the specification requires of your solution. Meet with your project partners and plan out the architecture of your solution. The more you think through and understand what needs to be programmed *before* you write code, the better. Design your project with independently and incrementally testable subsystems rather than saving all the testing for the end.

The second most common reason for spending too much time on a project is hunt-and-peck debugging; trying things at random, just to see if they work. If you find yourself in this position, step away from the keyboard and think about what is happening, or come see us during office hours.

There are many sources of help on which you can draw. Most questions can be submitted to the teaching staff and your fellow classmates via the course discussion forum. These will typically be answered the same day, often more quickly during working hours. However, some types of questions cannot be answered without seeing your project. If you have detailed questions about a program, speak to one of the teaching staff in person during office hours. Students are encouraged to help each other on course concepts (but not on the implementation of the projects). One of the best ways to make sure you understand a concept is to explain it to others. Keep in mind, however, that you should not expect anyone else to do any part of the project for you. The project that you turn in must be your group's own.

Many computing sites have consultants who are available to help you at the site. They are fine sources of help with questions regarding the computers and installed software (such as Unix and e-mail). However, they are not likely to be able to help you with questions about operating systems, C++, or specific errors in your programs.

6 Exams

There will be two exams during the semester: the midterm will tentatively be held Thursday 2/16 from 7pm to 9pm (pending room availability), and the final will be held on Monday 4/24, from 7pm to 9pm.

Unless a documented medical or personal emergency causes you to miss an exam, you will receive a zero for not taking an exam. If you anticipate conflicts with the exam time, talk to your professor as soon as possible, and at least one month before the exam date. The exam dates are given at the beginning of the term so that you can avoid scheduling job interviews or other commitments on exam dates. Outside commitments are not considered a valid reason for missing an exam.

7 Grading policy

Final grades will be based on the total points earned on the projects and exams. In addition, you must average a passing grade on the two exams to receive a passing grade for the class. Factors such as class participation may be used to adjust your final grade, especially if it falls on a borderline between two grades. The tentative breakdown is:

Project 1: 4%

Project 2: 14%

Project 2: 14%

Project 3: 14%

Midterm exam: 25%

Final exam: 29%

Incompletes will generally not be given. According to University policy, doing poorly in a class is not a valid reason for an incomplete. If you are having problems in the course, talk to an instructor as soon as possible.

8 Computers

You may use any x86 PC that runs Linux (version 2.6) and g++ (version 4.7.0). We recommend that you use the computers in the CAEN labs or Duderstadt Center, either directly or by logging in from another computer.

9 Attendance and discussion sections

You are expected to attend lecture regularly and to be at discussion section weekly. Discussion section meetings will typically involve active participation by discussion, group exercises, or question-and-answer sessions.

10 Lecture schedule

Week of	Monday	Wednesday	Readings	Notes
2-Jan		Intro	Ch. 1, 2-2.2, 4-4.1, 4.5	
9-Jan	Threads	Threads	Ch. 4.2, 4.6, 5-5.4	P1 out
16-Jan	MLK Jr. Day	Threads	Birrell	Group declaration deadline
23-Jan	Threads	Threads	Ch. 4.3-4.4, 5.5-5.7	P1 due, P2 out
30-Jan	Threads	Threads	Ch. 6-7	
6-Feb	Threads	Threads	Ch. 8	
13-Feb	Address Spaces	Midterm Review	Ch. 9	P2 due, Midterm 2/16 7-9pm
20-Feb	Address Spaces	No Class		P3 out
27-Feb	Spring Break	Spring Break		
6-Mar	Address Spaces	Address Spaces	Ch 2.3, 3-3.1, 10	
13-Mar	Address Spaces	Address Spaces	Ch. 3.3-3.6	
20-Mar	File Systems	File Systems	Ch. 11-12	P3 due, P4 out
27-Mar	File Systems	File Systems	Ch. 3-2, 13-14	
3-Apr	Networks	Networks	Arpaci-Dusseau (LFS)	
10-Apr	Dist. Systems	Dist. Systems	posted as needed	
17-Apr	Review			P4 due
24-Apr				Final exam: 4/24 7-9pm