Instructor: Nels Oscar

Office Hours: 1000-1200 TR in the Kelley Atrium and by appointment.


Description:
Introduction to operating systems using Unix as a case study. System calls and utilities, fundamentals of processes, threading, and interprocess communication.

Prerequisites:
- CS 261 – Data Structures
- (CS 271 or ECE 271) Computer Architecture and Assembly Language
- Experience programming in the C language.

You will use Unix and C extensively throughout this course, yet basic Unix and C will not be taught in this course. If you have not used Unix or C before, you will be expected to learn the basics on your own time. There are many Unix/Linux and C tutorials available on the web.

The specific variant of Unix that you will use in this class is Linux. If you are not sure where to start looking for tutorials on the web, I recommend using a search engine. You are also expected to have familiarity with common data structures, such as stacks, heaps, queues, linked lists, and hash tables, which are covered by the prerequisite class CS261 and its predecessors.

Course Objectives:
- Explain why multiprogramming is important for modern operating systems.
- Explain the general structure of a multiprogrammed operating system.
- Explain the purpose and operation of system calls.
- Write a program utilizing system calls.
- Write a program using a scripting language.
- Write a program that uses regular expressions to parse input data.
- Write a program that spawns processes and provides mutual exclusion for variables or other resources shared by the processes.
- Write a program that uses sockets to implement a client/server system.
- Explain how a common file system works, including structure, I/O operations, and security.
- Describe the memory organization of a typical process in a common operating system.
Grade Policy:

- Programming projects: 60%
- Quizzes: 20%
- Final exam: 20%

Projects:
All project must be submitted electronically by 11:59pm on the due date at the engr submission site.

Only a single late HW may be submitted. Late HWs have exactly one extra week, no more. Late work is penalized 5%/day.

All grading will be done via demo with your TA on the os-class server. We highly recommend that you do your development on a Linux operating system or at least test your assignments on os-class.engr.oregonstate.edu prior to submitting the assignment, a Linux machine provided specifically for use by CS344.

You can log on via SSH.

Standard campus computing policies apply: you must have a campus IP to directly log in, or log in via flip or nome.

Compilation of all assignments will be on os-class. Assignments which do not compile will receive a grade of 0. If it doesn’t compile on os-class with icc, it doesn’t compile.

All assignments will be turned via TEACH. All assignment submissions will use the following naming scheme: CS344_projx_y.tar.bz2, where x is the assignment number and y is your username. No other file formats will be accepted. NO .zip files. It is required that you make use of LaTeX to create the write-ups which will be part of each assignment. There are multiple packages for creating highlighted source code for LaTeX. No folder hierarchy is allowed inside the archive.

All work must be done individually unless you are specifically allowed to work in groups.

Quizzes:
There is no set number of quizzes. Quizzes will not be announced and will be administered during class time. These quizzes will check for understanding of recent course material.

Final Grade Assignment:
There will be no curve given in this course, though individual assignments will be considered for curving. The grading scale is as follows:

\[
\begin{align*}
90 & \leq A \leq 100 \\
80 & \leq B < 90 \\
70 & \leq C < 80 \\
60 & \leq D < 70 \\
00 & \leq F < 60
\end{align*}
\]
There will be opportunities on nearly every assignment for extra credit. Do not ask for extra credit beyond what is offered in class, as it will not be given.

Your final exam score plus your quizzes must have a weighted average $\geq 60$ in order to pass the course. You must pass the final exam to pass this class.

All programming projects must be submitted in order to pass the course. Students missing assignments at the end of the term will automatically receive an F grade.

Students who do not submit the final exam will automatically receive an F grade.

**Important Dates:**

This course is being taught during OSU Summer Session 3.

Refer to the Summer Session website for specific dates.

Most critically for our purposes this course starts on June 22 and finishes on August 14th.

**Academic Honesty:**

See the University and College policies.

Students are expected to do their own work, full stop. Do your own work.

Programming assignments present unique challenges for graders. It is often difficult for a grader to distinguish between legitimate help and plagiarism. Honesty is absolutely essential in order for learning to take place. It will form the foundation of your professional integrity in your career.

If you are having trouble with an assignment, **DO** discuss it with other students, TAs, the instructor, or anyone else who will listen, but don’t just have someone else tell you how to solve the problem! If other students ask you for help, **DON’T** just let them copy your work! It is possible to discuss problems without plagiarizing, and it is a valuable skill.

If you get help from, give help to, or work together with someone, you **MUST** (in the program header block) list that person as a collaborator and describe the help. Programs that are very similar will be subjected to review unless both programs indicate that they were produced collaboratively.

If you get help from printed or online sources, **DO** cite your references.

If you are found in violation of any of the above policies, whether you are the giver or receiver of help, you will receive a zero on the assignment or fail the course (Instructor’s discretion). The academic dishonesty charge will be documented and sent to your school’s dean and the Office of Student Conduct. The first offense results in a warning; the second offense results in an academic dishonesty charge on your transcript, a disciplinary hearing, and possible expulsion.

The bottom line is: Each student is expected to understand all aspects of the work s/he submits for credit.