CPS 310: Operating Systems (also ECE 353)
Fall 2020

Class Meetings

Prerecorded lectures (approximately 90 minutes per week) on Panopto
M 3:30 – 4:45 Activity session on Zoom
WF 3:30 – 4:45 Ghost class (optional except by announcement; instructor available)

Instructors

Michael Hewner
Jeff Chase

Website

https://www2.cs.duke.edu/courses/fall20/compsci310/

This course gives an introduction to systems concepts and operating systems. An operating system is software that controls some programmable platform for sharing resources and data. All operating systems must deal with core issues of protection, resource management, program environment and execution, coordination, and reliable state storage and recovery. Traditionally the course emphasizes classical operating systems topics: concurrency, facilities for storage, communication, and protection, kernel services and structure, architecture/OS interaction, distributed systems, and practical application of operating system concepts in real operating systems. We also touch on principles and topics that are important for understanding modern networked software and its performance.

Topics. A list of specific topics and related reading is available on the course web. We publish slides and other materials in a course git repository: see the course web. **Advanced topics**. In Fall 2020, the CPS 310 is synchronized and interleaved week-by-week with an advanced course (CPS 510) that runs concurrently in the same time slot on TTh. These courses share a git repository and Panopto folder. You will encounter CPS 510 material: you are encouraged to review it, but it is not required.

Preparation. You should be familiar with undergraduate-level computer architecture and consider yourself a strong student and a good programmer. You will program in C and C++. You should understand C pointers and basic data structures, e.g., lists, queues, stacks, hash tables, trees, graphs, DAGs.

Tools. Know the basics of the Unix command interface. We will use the git revision control system and your project code will be hosted in private repositories on github. *No public repositories please!* (We have an ongoing problem with Duke students publishing their code for prospective employers, which requires us to hunt them down. Please please don't do it.) The projects (labs) require an Ubuntu Linux environment, e.g., with Docker Desktop for your Windows or Mac machine. Labs are autograded by the AG350 service. Follow the directions on the course web.

Readings. There is no required textbook. We recommend the Web-available text. Operating Systems in Three Easy Pieces (OSTEP). The core material presented in lecture and slides/notes in the course repository. Students may improve their understanding by looking at one or more of the optional texts: see resources on the course website.

What to do. This semester we have no exams, but we add some new lab work in collaborative Zoom activity sessions and weekly assignments. Every week we have: lectures (pre-recorded), a synchronous activity session (M), activity submission on Sakai (T), a Sakai quiz on the week's material (F), and a lab, which is due via AG350 on the following Monday. Please contact us if you cannot attend the activity.

Projects / labs. Some of the labs are designated "advanced": it is possible to earn a grade of B- or lower without them. Both kinds of projects usually take 3-10 hours to complete, depending on the particulars and student background. Get help from an instructor or TA if you're stuck! There's no point spending 5 hours debugging one issue.

Lecture Quizzes	15%
Activity Submissions	15%
Regular Projects	50%
Advanced Projects	20%

Grading. Grades are based on a point score weighted as in the table. The weeks are equally weighted.

*Lecture Quizzes/Activity Submissions***.** Lecture quizzes are short quizzes design to ensure you watched the weekly lectures. Activity submissions are just submitting what your team worked on in the activity to ensure you participated. We will drop the lowest grade of both these categories, because we know that sometimes there are unexpected internet blackouts or other issues. Please finish these before the deadline and save your dropped grade for really unexpected problems.

*Regular/Advanced Projects***.** Regular projects are the critical required projects in the course and make up a large portion of your grade. Advanced projects are additional assignments for students who want to dig deeper—they may be "harder" and they are worth less. To get an A in 310 you must succeed on at least some of the advanced projects, but for a C you don't need to do any.

Late work. Graded work has deadlines. To keep things fair no extensions can be granted to individuals or groups. We may occasionally grant extensions to the entire class, or grant "free passes" for late days that all groups may "spend" as they choose. Late work receives a penalty of at least 20%, or more, depending on circumstances. It is much better to do the work and hand it in late than to receive a zero on any assignment.

Assistance. We will provide online assistance through Piazza: see the course web. Please post your questions there. Piazza etiquette: check for answers before posting, keep posts clear and tight and include all relevant information, check your civility, don't reveal too much in public posts. In addition to two TAs we have UTAs to assist you with the projects. Also, you will find us to be generally welcoming—email or chat us on Teams for a live consultation if needed.

Policy on collaboration for CPS 310. Collaboration on lab/project work is acceptable. We encourage social contact in these days of COVID. In particular, you are encouraged to share your knowledge, your understanding of the assignment, and your experience with "sticking points". However, sharing of code across groups is not allowable. Any work you turn in must be your own, and you may be called upon to explain (alone) your choices and approaches in more detail. You may incorporate public software into your assigned lab work and course project to a reasonable extent, but not so much as to undermine the educational purpose and spirit of the project. ***You must acknowledge any sources of your words, ideas, and software when they are not your own, and you must disclose (in advance, in a README.txt file in your submitted code, without any specific request) any sources you consulted.*** Unacknowledged use of another's code is cheating. Receiving assistance from any external source on quizzes or exams is cheating.

In particular, it is forbidden to incorporate code from students who took the course in a previous semester, or from students at other schools using similar projects. Please do not place your code in open repositories where they can tempt other students. If you discover one please let us know about it.

For practical reasons, we choose to treat code sharing among groups *in the same semester* as a public health problem rather than an offense under the university's academic dishonesty policy, to the extent that the assistance is acknowledged. Your intent is to learn. Our intent is to help you learn and to avoid creating a climate of secrecy and fear around ordinary assistance or use of example code from documentation or other common sources. But do not share code across groups. All students should understand that we have software that flags copied code with a high degree of certainty and precision. (The tools do not differentiate the makers from the takers.) Sharing that is "over the line" may result in a downward grade adjustment, according to our discretion.