CPS 310: Operating Systems (also ECE 353)
Spring 2017

Class Meetings
WF 1:25 – 2:40 in LSRC B101 (Love Auditorium)
M 1:25 – 2:40 in LSRC B101 (TA recitation)
Instructor
Jeff Chase
Office hours: [T 11:00-12:00, F 3:00] in D306 LSRC, or by appointment, or try a drop-in.
Teaching Assistant
Christopher Streiffer
Website
http://www.cs.duke.edu/~chase/cps310

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 ecosystems and their performance.

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

Tools. You should know the basics of the Unix command interface: see the CSL tutorials. We will use the git revision control system: project code will be submitted via gitlab.oit.duke.edu. *No public repositories please!* You can do some projects natively on a mac (just install Xcode from Apple), but we strongly recommend using Ubuntu Linux under VirtualBox to ensure compatibility with the grading software. That should work on any reasonable machine you might have. You can also use login servers in the CS department (*login.cs.duke.edu*): let us know ASAP if you need an account created.

Readings. There is no required textbook. We will use the Web-available text. Operating Systems in Three Easy Pieces (OSTEP). The core material will be presented in lecture and notes (including almost 1000 powerpoint slides!). Students may improve their understanding by looking at one or more of the optional texts: see resources on the course website.

Base workload. In addition to the readings, there four assigned projects/labs (done in teams of 2-3), two midterms, and a final exam. We may also have in-class quizzes on Sakai to supplement the exams. Students report spending at least 10-15 hours on each lab. Many groups spend more time and a few spend much more time: as with all software development, the more bugs you create, the more time you will spend fixing them. You may choose your own groups: please follow guidelines for group submission on the course web. Here are the dates for Spring 2017:

- Jan 31: (Tuesday) Heap manager
- Feb 17: (Friday) Midterm #1
- Feb 28: (Tuesday) Threads lab
- March 28: (Tuesday) Black hat lab
- April 10: (Monday) Midterm #2
- April 18: (Tuesday) Distributed consensus lab
- May 5: (Friday) Final exam (9:00 AM - 12:00 PM)

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 between 15% and 60%, 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. In addition to the TA we have seven UTAs to assist you with the projects. The UTAs will post office hours on Piazza.

Attendance. Attendance or lack of attendance in class/recitation is not recorded. You do not need permission to skip class. The lectures are recorded and are available through Panopto. However, my experience is that students who come to class and are engaged tend to do better in the course.

Grading. The semester grade is determined from your exam grades (50%) and project work (50%). I make adjustments of up to half a letter grade for subjective factors, including trends of improvement or decline and project group dynamics. All students in a project group receive the same grade for the group's projects, but each group member should be familiar with all aspects of the group's solution (I often ask about your solutions on exams). Grades are curved, but I give equivalent grades for equivalent performance. In the past about/up to 40% of the class have received grades of A- or above, with B grades assigned below a natural break. Some students have more difficulty (or put in less effort) and receive lower grades. If you are struggling, please visit me (early!) so we can figure out how to get you on track. As a rule, students who stay engaged and complete the work receive passing grades. Additional information about grading policies, projects, and exams is available on the course web.

Topics. A list of specific topics and related reading is available on the course web.

Policy on collaboration for CPS 310. Collaboration on lab work and project work is acceptable. In particular, you are encouraged to share your knowledge, your understanding of the assignment, and your experience with "sticking points". Discussion on Piazza is encouraged. 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 file in your submitted code, without any specific request) any sources you used.*** 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. Do not place your code in open repositories where they can tempt other students.

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. The intent is 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.) Code sharing may/will result in a downward grade adjustment, according to our discretion.