Instructor

Bruce Molay (molay@fas.harvard.edu), (617)864-8832.

Purpose/Content

Csci-e28 explains the structure of the Unix operating system and shows how to write system and network programs. It is appropriate for students who want to learn how to write system software for Unix or for students who want to learn about the structure of a multi-tasking, multi-user operating system. The course covers the details of the file system, terminal and device input/output, multi-tasking, interprocess communication, video displays, and network programming. Theory is presented in the context of how Unix implements the ideas. By the end of the course, students should be able to figure out how most Unix commands work and know enough about the system to draft their own version of most of them.

Preparation

You should be able to program in C or C++. You should be comfortable with pointers, structs, dynamic memory allocation, linked lists, and recursion. You do not need to have programmed in C for Unix. If you know C++, you need to write in the C subset of C++. Students are already expected to be comfortable with designing, coding, and debugging programs of modest complexity while employing good programming style, structured techniques, and employing appropriate data structures as appropriate. Familiarity with Unix is helpful but not essential.

Classes

Classes are Wednesdays, 7:40-9:40PM ET online using Zoom. Lectures present ideas in the context of specific problems and Unix commands. Careful review of sample programs will be used to demonstrate principles and focus discussion. Many of the programs will be from the text; read before lecture, bring questions.

Reading

Understanding Unix/Linux Programming by Molay is the main text This book follows the course closely. Two other texts are not required, but are helpful additions: *Advanced Programming in the Unix Environment* by Stevens is more encyclopaedic - has all the info and is an excellent reference. *Linux Application Development* by Johnson and Troan provides supplemental information about programming for Linux.

Required Work

A final exam and six programming assignments. The assignments are spaced evenly through the term. Most assignments build on or follow examples and ideas presented in class. Grades are based on a final exam and the programming projects. The weighting is roughly 35% exam, 61% for projects, 4% for class participation. For details on assignments, see the Assignments page on the course web site (cscie28.dce.harvard.edu/~dce-lib215).

Final Exam

The final format is to be determined: either proctored at home with Canvas or proctored in person on paper.

Course System, VPN

The course machine is cscie28.dce.harvard.edu. Connect using ssh over the Harvard VPN. Detailed instructions for the VPN can be found here: https://harvard.service-now.com/ithelp?id=kb_article&sys_id=f1766696db1e94184415 60fdd39619ef

Accounts

Your username on the E28 server is your Harvard NetID. You need a *Harvard Key* to get a NetID. Claim your *Harvard Key* at https://key.harvard.edu/ . For details about claiming your key, visit: https://extension.harvard.edu/for-students/support-and-services/computer-and-e-mail-services/

You can find your NetID at https://key.harvard.edu/manage-account. Accounts will be available one week before classes start.

Help

Review Sections	Online meeting one hour each week at a time to be determined.
Office Hours	Online with Zoom, times to be arranged
Ed	Use Ed discussion on Canvas to send questions to staff and to start and participate in class discussions.
web page	http://cscie28.dce.harvard.edu/~dce-lib215/

Info Sheets/Scheduling

Please go to http://cscie28.dce.harvard.edu/~dce-lib215/infoform to tell us your preferences for section and office hours time. Do this now.

Accessibility

The Extension School is committed to providing an accessible academic community. The Accessibility Office offers a variety of accommodations and services to students with documented disabilities. Please visit www.extension.harvard.edu/resources-policies/resources/disability-services-accessibility for more information.

Academic Integrity/Plagiarism

You are responsible for understanding Harvard Extension School policies on academic integrity (www.extension.harvard.edu/resources-policies/student-conduct/academic-integrity) and how to use sources responsibly. Not knowing the rules, misunderstanding the rules, running out of time, submitting the wrong draft, or being overwhelmed with multiple demands are not acceptable excuses. There are no excuses for failure to uphold academic integrity. To support your learning about academic citation rules, please visit the Harvard Extension School Tips to Avoid Plagiarism (www.extension.harvard.edu/resources-policies/resources/tips-avoid-plagiarism) , where you'll find links to the Harvard Guide to Using Sources and two free online 15-minute tutorials to test your knowledge of academic citation policy. The tutorials are anonymous open-learning tools.

Details Academic conduct

Unless otherwise stated, all work submitted as part of this course is expected to be your own.

You may discuss the main ideas of a given assignment with other students (provided that you acknowledge doing so in your solution), but you must write the actual solutions by yourself. This includes both programming assignments and other types of problems that we may assign.

Prohibited behaviors include:

- Copying all or part of another person's work, even if you subsequently modify it
- Viewing all or part of another student's work
- Showing all or part of your work to another student
- Consulting solutions from past semesters, or those found in books or on the Web
- Posting your work where others can view it (e.g., online)
- Receiving assistance from others or collaborating with others during an exam, or consulting materials except those that are explicitly allowed.

If we believe that a student is guilty of academic dishonesty, we will refer the matter to the appropriate administrative committee. Penalties for this type of behavior are typically severe.

Generative AI

Course Goals: The goals of CSCI-E28 are to help you understand the Unix/Linux system API and to improve **your** programming and design skills. In the same way that using Google Translate to do assignments for a course in French language and culture prevents students from actually learning French language and culture, using Chat-GPT or other generative AI system to produce syntax, algorithms, and problem-solving prevents **you** from actually learning syntax, algorithms, and problem-solving.

In order **to achieve these goals**, we expect students to practice syntax, algorithm design, and problem solving. We expect that all work students submit for this course will be their own. We specifically forbid the use of ChatGPT or any other generative artificial intelligence (AI) tools at all stages of the work process, including preliminary ones. Violations of this policy will be considered academic misconduct. We draw your attention to the fact that different classes at Harvard could implement different AI policies, and it is the student's responsibility to conform to expectations for each course.

Attendance/Participation

Students are encouraged to attend class during the live presentation and to participate by asking and answering questions. People who cannot attend class may participate by attending office hours, review sections, and on the Ed discussion site.

Credit/Work

Students enrolled for graduate credit will do additional software design/planning documents for each project.