CSE 374: Programming Concepts and Tools

Instructor: Eric Mullen (emullen@cs.washington.edu), office CSE 218
Lectures: 11:30-12:20 in BAG 260

Communication: While you can email me, I highly encourage you to email the course staff instead (which includes me) at cse374-staff@cs.washington.edu. This will get a faster response, and if you email me, I will likely reply to tell you to email the staff. Please don't email individual TAs without an extremely good reason.

Goals: Successful course participants will:

- Gain a basic familiarity with the Linux operating system and toolchain.
- Develop the skills to automate common computing tasks such as file manipulation and string processing
- Internalize C-level programming and obtain beginning proficiency in C programming
- Learn the basics of programming tools such as debuggers, profilers, compilation managers, and version control
- Learn core software-engineering practices regarding specification and testing
- Understand the basic issues and pitfalls of shared-memory concurrency
- Learn how to acquire additional information and skills independently

Prerequisite: CSE 143 (i.e., intermediate programming in Java, including basic data structures like lists, queues, and binary search trees)

Grading and Exams:

- 55% - Homework assignments, approximately weekly
- 15% - Midterm exam
- 25% - Final exam
- 5% - Class participation, citizenship, and other

In principal, all homeworks contribute equally to the 55%, but larger programming projects will be weighted somewhat more than others. Percentages are tentative and may be adjusted.

Extra Credit Policy:

- Extra credit is designed to have little (but some) impact on your grade whether you do it or not. Not doing extra credit will not lower your grade -- regardless of how many other students attempt it.
- Extra credit is designed to be challenging and an opportunity for people with extra time to work on something optional.

Therefore:

- Expect an extra-credit problem to be worth much less than it should be based on how difficult it is. For example, if a homework is graded out of 100 points, the extra credit may be worth a maximum of 4 points with no chance for partial credit.
- You should not attempt the extra credit until you have finished the rest of an assignment. We will ignore extra-credit work on assignments that are not at least "almost perfect" otherwise.

Late Policy: Deadlines will be given with each assignment. These deadlines are strict. It is exceedingly unlikely that skipping class or being late to class because of homework is in your interest. For the entire quarter, you may have four "late days". You are strongly advised to save them for emergencies. You may not use more than two for the same assignment. On group projects you may only use late days if all members of the group have them available, and all members of the group will be charged for each late day used. They must be used in 24-hour (integer) chunks. This policy may not be the same as in your other classes. You are responsible for understanding it if you choose to submit late work.

Academic Integrity: Any attempt to misrepresent the work you submit will be dealt with via the appropriate University mechanisms, and your instructor will make every attempt to ensure the harshest allowable penalty. The guidelines for this course and more information about academic integrity are in a separate document. You are responsible for knowing the information in that document.

Text: There are two books listed for the course. The Pocket Guide is "required" - it is a concise source of information and provides a useful reference to Linux at the level you need for CSE 374 (of course you should *not* feel responsible for memorizing all of the details in it). The Kernighan & Ritchie book is the classic reference on C from the people who invented the language. It is "optional", but has useful explanations and examples beyond the information presented in class or reference information available online.

- *Linux Pocket Guide* by Daniel Barrett, O'Reilly, 2nd ed., 2012. If you have a copy of the first edition that should work as well.
- *The C Programming Language* by Brian Kernighan and Dennis Ritchie, Prentice-Hall, 2nd ed, 1988.

See the Linux and C resource pages on the course web site for links to additional information.

Advice: This course will expose you to a tremendous number of tools, concepts, and issues. Be warned:

- The unease from using new tools may drain your energy because you will constantly learn new tools. Don't yield to the temptation to avoid learning new tools because it may seem "quicker" to finish assignments in other ways - that misses the whole point of the exercise.

- The lectures will not teach the tools with enough detail to do the homework. Rather, they will give you the concepts behind the tools and enough information to point you in the right direction.
- You will not master everything in 10 weeks, but you will learn enough to continue increasing your proficiency and more easily learn what you need in the future.
- If you are spending an enormous amount of time on an assignment, you are likely missing a key concept. Spending more time "fighting through it" or randomly Googling for fragments to try is not effective; use yourself and the course staff to determine what you are missing.

Approximate Topics (subject to change): We will do almost all of the following, but not necessarily in this order.

1. Files, processes, and shells (6 classes)
    a. Command-line utilities
    b. File editing
    c. Shell scripting
    d. String processing; regular expressions
    e. Web basics (http, html)
2. C Programming (6-7 classes)
    a. Memory model
    b. Pointers
    c. Arrays
    d. Manual resource management
    e. The preprocessor
    f. Idioms for safe programming
3. Programming tools (6 classes)
    a. Debuggers
    b. Profilers
    c. Linkers
    d. Compilation managers
    e. Version-control systems
4. Software-engineering issues (2-3 classes)
    a. Multiperson programming
    b. Specification
    c. Testing
    d. Code-reuse patterns
5. C++ (2-3 classes)
    a. C with classes and objects
    b. Other differences from C
6. Concurrency (2-3 classes)
    a. Threads
    b. Races and deadlocks
    c. Locks and Transactions
    d. Message-passing