CS 146 — Software Tools and Systems Programming in Unix and C

Instructor: Wayne Hayes, whayes@uci.edu, DBH4092

Goals: This course is an in-depth study of principles and concepts embodied in modern Internet-connected, multiuser, multitasking operating systems (MICMMOS, :-)), including shells, filters, pipelines, programmability and scripting, extensibility, concurrent processing, and interprocess communication. Several integral tools and utilities are presented. Unix is used to provide concrete examples, focusing on the file system, processes, shells, and various tools including awk, diff, file, find, ftp, grep, make, man, nslookup, sed, sort, tar, traceroute, uniq, vi, which, zip, etc.

The course provides a solid conceptual and experiential basis for subsequent work in the programming, system administration, and effective general use of the Unix operating system. This conceptual basis for subsequent work is more generally applicable to other MICMMOS's and to later courses which study or use them.

Required Texts: We will use the following *classic* textbooks. Despite their age, in this instructor's opininion they are second to none for *succinctly* teaching all the basics you need to become a proficient programmer (**not sysadmin!**) on Unix systems. They both assume that *you have significant previous programming experience*. However, they can be hard to come by. I have seen both on Amazon (often used, which is fine) for $20 or less.

- *The UNIX Programming Environment.* Kernighan & Pike.
- *The C Programming Language*, 2nd or late ed. Kernighan & Ritchie.

Other useful texts: If the above don't provide enough introductory material, try these:

- (for true beginners) *UNIX Command Line: A Complete Introduction.* William Shotts Jr..
- (more seasoned users) *Linux Command Line and Shell Scripting Bible.* Blum & Bresnahan.
- Kelley & Pohl, *A Book on C.*
- Abrahams & Larson, *UNIX for the Impatient*–a big, big book. Excellent reference. Bad for a text.
- Kernighan & Plaugher, *The Elements of Programming Style*–a classic, generally good advice for Unix & C programming.
- Weiss, *Efficient C Programming*–some more advanced topics, much good advice
- Garfinkel, Weise, Strassmann, *UNIX Hater's Handbook.*

Grading

Assignment/test/exam	Value	out	due	Duration
#1	5%	Week 1.1	Week 2.1	1 week
#2	5%	Week 2.1	Week 3.1	1 week
#3	10%	Week 3.1	Week 5.1	~2 weeks
#4	10%	Week 5.1	Week 7.2	~2.5 weeks
Midterm	15%	Week 6.1	in class	~50 minutes
#5a (project-A)	5%	Week 7.2	Week 9.2	~2 weeks
#5b (project-B)	10%	Week 7.2	Week 10.3	~3.5 weeks
	[absolute deadline]		Week 11.3	~4.5 weeks
Exam	40%			

To pass the course you must pass the exam. (But it's not hard; blank answers receive 1/3 credit!)

Late Policy for assignments:

```
        - assignments due at the START of class
        - excuses will be listened to.  Good ones will be accepted.

                received                          penalty (out of 100%)
                ========                          =========
on time before START of class        0%
                up to 1 business day  (24h) late     8%
                up to 2 business days (48h) late    16%
                up to 3 business days (etc) late    32%
                up to 4 business days late          64%
                up to 5 business days late         128%
```

Regrading requests: All requests for regrading must be submitted in writing with an explanation of why you think you should get more marks (email is fine, though you might consider marking up a PDF or photo of what you're referring to if it's not obvious). Comparisons with the grades of other students in the class are allowed but in this case both assignments must be submitted for regrading. In any case, the **entire** assignment will be regraded; thus, you risk losing marks in other places where the grader may have been generous, even if your case has merit.

Academic dishonesty: **Academic dishonesty** will not be tolerated in any form. You may discuss ideas with your classmates (and others), but no written notes should be taken away from such discussions. All work you hand in must be your own. This course strictly adheres to all relevant University and ICS policies. It is each student's responsibility to be aware of these policies. To this end, all students are advised to (re)read the UCI Academic Senate Policy on Academic Honesty, noting in particular that any single incident of student academic dishonesty in this course is sufficient to merit a failing grade in the course with a letter of explanation being placed in the student's file. The Academic Honesty Policy for Information & Computer Science also applies to this course and deals explicitly with with course work involving computers.

If you are unsure whether certain behavior is acceptable, ask before you engage in it.

Why Unix?

- "The UNIX system can't last forever, but systems that hope to supercede it will have to incorporate many of its fundamental ideas." Kernighan & Pike, *The Unix Programming Environment* (Prentice-Hall, 1984)
- Those who do not understand Unix are condemned to reinvent it—poorly.
- Unix gives you just enough rope to hang yourself—and then a couple of more feet, just to be sure.
- Unix is the answer, but only if you phrase the question very carefully.
- Unix is user-friendly. It's just very selective about who its friends are.

By the end of this course, *Unix* will be your friend.