CSCI 352 - UNIX Software Development
Winter 2016

Time and Place: MTWF 10am AW 203

Instructor: Phil Nelson

Office/Phone: CF471, 650-3035

Office Hours: 1:30PM MTWF, others by appointment

E-mail: My e-mail address is phil.nelson@wwu.edu. When you send me e-mail, please use plain text (no HTML) messages and include “CSCI 352” in the subject of your e-mail.

Web Access: I have information for this class on the web. As they are assigned, I will put a copy of each assignment on the web. See the page http://facultyweb.cswwu.edu/~phil. Other information will be provided via the web. If you believe something is missing from the site, please e-mail me requesting the information be posted. (This is different than what is currently linked on the department web site.)

(Available at http://proquestcombo.safaribooksonline.com/9780321638014)

References: Oram, Talbott, Managing Projects with Make, Prentice-Hall, O’Reilly, 1991. (Out of print.)
(Available at http://proquestcombo.safaribooksonline.com/0596006101)
(Available at http://proquestcombo.safaribooksonline.com/0596003307)
(Available at http://proquestcombo.safaribooksonline.com/9780133085129)
Other Reference: See the web. Also, a student ACM membership gives you access to ACM’s “Books24x7” which contains a few UNIX introduction books.

Description: This class is intended to teach you about software development in the UNIX environment, both user level and system level development. It also covers a few of the UNIX tools available to ease your work load.

Course Outcomes: On completion of this course, students will demonstrate

- A thorough understanding of the design and development of systems and programs in the UNIX environment.
- A Basic understanding of the fundamental concepts of the UNIX operating system, its standards and implementations.
- A thorough understanding of the UNIX system data files, process environment and system calls.
- A basic understanding of shell programming, UNIX development tools and system utilities.
- The ability to design and implement substantial applications and utilities for the UNIX environment.

Graded Work: The graded work will be 3 tests and 6 assignments.

Tests: The tests are scheduled for Wednesday, January 27, Monday, February 22 and the final on Monday, March 14, 8AM.

Each test is worth 17% of your final grade and covers 1/3 of the class. The “final” is not comprehensive.

Minimum Points on Tests: To pass this class, you must earn at least 50% of all test points.

Assignments: Assignments will be worth 49% of your grade and will be programming in the UNIX environment. Assignments will be worth a different number of points and will contribute to “total” for the assignments. The assignments are scheduled to be due on January 12, January 20, January 29, February 12, February 26 and March 11.

They will be worth respectively 100, 125, 200, 200, 250, and 325 points for a total of 1200 points.
Coding Standards: All written assignments are required to follow the coding standards as listed on the web site at the URL http://facultyweb.cs.wwu.edu/~phil/classes/coding.pdf. Not following these standards may cause point loss.

Environment and Grading: All your assignments must work correctly in the Linux environment provided by the department. For assignments 2 through 6, I will provide you access to my grading script before you turn in your program so you can know how well your program is working before you turn it in. More details will be provided as to how to get to the grading scripts and run them, how to turn in your assignments and so forth.

Programs: Programming assignments are given to help you learn about UNIX programming. Assignments are to be written in C, not C++. The operating system will be a UNIX variant. Don’t do your programming under Windows and then move your stuff to UNIX for the final testing. Don’t even use visual studio as your editor. (The only exception allowed to use Windows is if you use Cygwin and it should be installed with the UNIX default file formats. Your program must still run on a “real” UNIX machine! You also should use vi, emacs, or other editor available with Cygwin.) Do not use an IDE (integrated development environment) like Xcode or eclipse for program development. Use an editor and command line tools for your work. For each assignment, you will make your source code available for testing by me. You will be told later how to do this. Please do not e-mail your programs to me! You can assume that a program that is not working in any way will receive 50% or less of the points possible for that program.

Final Assignment: The final assignment will be due at the last class period of dead week, March 11, 2016.

Late Work: Work is due at the beginning of class on the day due. Work will be accepted up to TWO meetings of the class late and will be worth 75% of the original value. (For example, if the assignment is due on Thursday, the second meeting of the class would be the following Monday.) Work later than two class periods will be worth nothing. A late final assignment is worth 75% of the original up until the start of the final test.
Grading: Grading is done by a percentage of the top score. The following is an example grade scale.

- **A:** 100% – 90%
- **B:** 89% – 80%
- **C:** 79% – 65%
- **D:** 64% – 50%

**Collaboration:** Each student *MUST* do their own programming. Original work is required. You should not see the source code of any other student, current or past, on this project. You may discuss problems using diagrams on scratch paper, but you should not see source code. Even helping a fellow student debug their program so that source code is seen should be avoided. Students having problems should e-mail me or visit me in my office.

**Cheating:** Is (obviously) not allowed. If you do cheat and are caught you will receive an F as your grade for the class. This includes *ALL* students knowingly involved in any cheating event. Not properly protecting your source code may be considered knowingly involved. If you give your password to your friend or allow access to your files or a machine on which your sources are stored, this can be considered knowingly involved. I use mechanical means to compare student programs, not only all students this quarter, but from students who took this class in the past. These comparisons are used to raise the possibility of cheating, but all decisions about cheating will be made by me after inspecting the programs of all students involved. *NOTE:* Making your source files for this class available for public, unprotected access will be considered cheating and may even get you an F for this class *AFTER* a passing grade has been reported and you have graduated.

**Repeating the class:** If you are repeating this class, I require you to throw away all your source code to all assignments and do them without reference to your previous work. Reusing your old assignments will be considered cheating.

**Topic Outline:** This does not give the exact order of the topics.

- Operating Systems and System Calls (Ch 1)
- Library Calls
- Manual pages
• Shells, fork, exec, wait
• History of UNIX
• Basic Unix (Ch 1)
• File I/O (Ch 3)
• gdb
• make (Make book)
• cvs (CVS book)
• Files and Directories (Ch 4)
• C Standard I/O Library (Ch 5)
• System information (Ch 6)
• Pipes (Ch 15.2, 15.3)
• Signals (Ch 10)
• Process Environment (Ch 7)
• Process Control (Ch 8)
• Process Relationships (Ch 9)
• Terminal I/O (Ch 18) (partial coverage)
• Advanced I/O (Ch 14) (partial coverage)
• Daemon Processes (Ch 13) (partial coverage)
• Utility programs
• Free and Open Source Software