CSCI 347 - Computer Systems II  
Summer 2022

**Time and Place:** MTWR 10am-11:20am, CF 316

**Instructor:** Phil Nelson

**Office/Phone:** CF471, 650-3035

**Office Hours:** 11:20am-11:50am MTWF, others by request.

**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 347” 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 [https://facultyweb.cs.wwu.edu/~phil](https://facultyweb.cs.wwu.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.

**Canvas:** All assignments must be turned in on canvas and with sources committed and pushed as requested in the assignments to gitlab.


Your text is avilable on-line via Safari-on-line. It costs a bit for access but may be less than an actual book. Your text URL is [https://learning.oreilly.com/library/view/advanced-programming-in/9780321638014](https://learning.oreilly.com/library/view/advanced-programming-in/9780321638014)

**References:** For the remaining tools, the URLs start with “https://learning.oreilly.com/library/view”


The URL tail is “managing-projects-with/0596006101”
The URL tail is “unix-power-tools/0596003307”

The URL tail is “practical-guide-to/9780134774626”

Other Reference: Safari on-line has a lot of other UNIX books.

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

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

- A thorough understanding of development in the UNIX environment
  - The ability to design and implement substantial software project in the C programming language.
  - Basic understanding of shell programming, UNIX development tools, and system utilities.
- A basic understanding of operating systems including file systems resources and system calls.
- A basic understanding of interprocess communication including pipes, shared memory and messages.
- Strong understanding of problems and techniques in concurrent programming.
- A thorough understanding of the purpose and use of semaphores, monitors, and rendezvous in concurrent programming.

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

Tests: The tests are scheduled for Tuesday, July 5, Monday July 18 and on Thursday, July 28. For each test you will have 85 minutes in which to take the test. Each test is worth 15% 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 55% 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 June 27, July 5, July 11, July 19, July 25 and July 28.

Coding Standards: All written assignments are required to follow the coding standards as listed on the web site at the URL https://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 some assignments, 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 and concurrency. 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 the class on Friday, July 29.
Late Work: Work is due at the beginning of class on the day due. That means that it is due at 11am on the day due. Late work will be accepted up to TWO days for which the class was scheduled to meet and will be worth 85% of the original value. (For example, if the assignment is due on Wednesday, the next “class day” is Friday and the second “class day” would be the following Monday on which the late work is due.) Work later than two class days will be worth nothing. The final assignment will not be accepted late.

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. Programming at the same time next to another student while actively writing code together also should be avoided. Students having problems should e-mail me or visit me in my virtual office. (See above.)

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. (The University’s policies and procedures regarding academic honesty are published in the catalog, Appendix D.)

**Repeating the class:** If you are repeating this class or taken CSCI 347 from another professor, 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)
- 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)
- Processes (Ch 7, 8 and 9)
- Concurrency (Ch 11 and 12)