CSCI 347 - Computer Systems II
Spring 2026

Time and Place: MTWF 9:00am, CF 314

Instructor: Phil Nelson

Office: CF473

Office Hours: 1:00-2:00 pm TW, 2:00-3:00 pm MF others by request. (e-mail is a good way to request a meeting outside of regular office hours.)

E-mail: My e-mail address is phil@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 and canvas. As assignments are ready, I will put a copy of each assignment on canvaqs. The class web page is:
`https://facultyweb.cs.wwu.edu/~phil/classes/s26/347`. 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: You will be using canvas to turn in assignments. **Nothing submitted to canvas means you did not turn in the assignment.** Canvas will determine late submissions. Actual source code will be in a CS gitlab project that will be created as part of your first assignment.

"Text": Stevens and Rago, *Advanced Programming in the UNIX Environment, 3nd Ed.*, Addison Wesley Longman, Inc., Reading Mass, 2005, ISBN-13 978-0-321-63773-4.

References: Links are on the class website at:
`https://facultyweb.cs.wwu.edu/~phil/classes/s26/347`.

Oram, Talbott, *Managing Projects with Make*, Prentice-Hall, O'Reilly, 1991. (Out of print.)

Mecklenburg, *Managing Projects with GNU Make, 3rd Ed*, O'Reilly, 2004, ISBN-13 978-0-596-00610-5.

Powers, Peek, O'Reilly, and Loukides, *UNIX Power Tools, 3rd Ed*, O'Reilly Media, Inc., 2002, ISBN-13 978-0-596-00330-2.

Sobell, *A Practical Guide to Linux Commands, Editors, and Shell Programming, 4rd Edition*, 2012, ISBN-13 9780134774626.

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, an 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. Attendance will also be part of the final grade formula.

Tests: The tests are scheduled for Friday April 24, Friday, May 15, and the finals on Wednesday, June 10 at 8am.

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 53% 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 April 10, April 22, May 6, May 20, May 27, and June 5.

Attendance: Due to low attendance in the recent past, I have now instituted an attendance requirement. This will be worth 2% of your final grade. There are 38 class periods in the quarter. Two of those are tests. There are then 36 class periods where attendance will be taken. On time gives you 2 points, late up to 10 minutes gives you 1 point and after 10 minutes you get no points for attendance.

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 or WSL and it should use 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 or WSL.) **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!** If you have questions about your program, commit them to your gitlab repository and push them. That way I can get them and look at your code without you e-mailing them 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, June 5, 2026.

Late Work: Work is due *at the beginning of class on the day due.* That means that it is due at 9am 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 75% 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. A late final assignment is worth 75% of the original up until the start of the third exam during test week.

Grading: Final grading is done by a percentage of the top score. Canvas will show your final percent of points based on the weighted scores as described above. At the end of the class, this weighted percentage will be used to assign grades. 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. You should not work together in designing solutions even with pseudo code.

Even helping a fellow student debug their program so that source code is seen should be avoided. Programming at the same time "sitting" next to another student (either in person or virtually) while actively writing code together also should not be done. Students having problems should e-mail me or visit me in my 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.)

AI: Using AI to help do or do your programming is a form of cheating. You have crossed the "line" if you upload the assignment description to an AI platform. Acceptable AI use is having AI generate sample multiple choice test questions so you can study for tests and asking for help understanding a concept. Any request for code to sovle a problem you are having is not acceptable use.

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 this quarter's assignments without reference to your previous work. Reusing your old assignments will be considered cheating.

Western Syllabus Policies: For generic syllabus policies of Western, visit `https://syllabi.wwu.edu`.

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

- Quick history of UNIX
- 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)