

CS 306 – Linux/UNIX Programming – Spring 2012

MWF 3:00–3:50 p.m., Parkinson 107

See the course web page for more information and resources: <http://www.cs.siu.edu/~cs306>

Professor: Dr. Norman Carver, Faner 3121, phone 453-6048, email carver@cs.siu.edu.
Office hours: MWF 1:00–3:00pm (or by appointment).

TA: Jason Fairfield, Faner 3129, phone 453-6036, email pegasusjf@siu.edu.
Office hours: Tues, Thurs 10:00am–12:00pm

Required Text: (choose either of the following)

- (1) *GNU/Linux Application Programming* (2nd ed) by M. Tim Jones.
- (2) *Beginning Linux Programming* (4th ed) by Matthew and Stones.

Workload: labs and homework (required for exam admittance!), three exams (25% each), final exam (comprehensive but optional, 25%).

Dates: vacations: 3/12–3/16; last drop date: 3/18 (online);
final exam: Tues., 5/8, 5:50–7:50pm.

Goals of the Course

- Become comfortable using basic Linux/UNIX commands and utilities.
- Introduce the C language and get experience programming in C.
- Learn how to develop software on and for Linux/UNIX systems.
- Become familiar with important Linux/UNIX library functions and system calls.
- Introduce asynchronous and concurrent (multi-process) programs.
- Introduce network programming.
- Lay a foundation for an operating systems course.

Overview of Course Topics*

1. Introduction to operating systems and Linux/UNIX
2. Effective Bash shell (CLI) usage
3. Introduction to the C language
4. Linux/UNIX programming and development tools
5. Systems programming
6. Concurrent programming
7. Network programming

*See the course web page for a more detailed outline of topics to be covered, along with the relevant textbook sections, slides, and handouts.

Prerequisites, Attendance, and Textbooks:

- The CS Dept *prerequisite* will be enforced (CS 220 w/C or better, or equivalent).
- You are *not* required to be familiar with the C language or with the Linux OS.
- You are expected to be familiar with programming in at least one C-family language (Java, C, C++), through the level of a standard CS2 course (i.e., data structures).
- *Attendance* at the lectures is considered an *important* part of this course. Frequent absences will almost certainly impact the quality of your work and will definitely make the instructor and TA less willing to help you outside of class (since you will probably end up asking about issues that were already covered in class).
- Having access to one of the textbooks is considered a *requirement* of this course, and you are *expected to read the assigned sections in one of the course textbooks*. While all of the course topics will be covered to some degree in lectures, the lectures cannot cover everything you should learn from this class due to time limitations. You may be tested on assigned material in the texts even if it has not been presented in lectures.

Grading:

- A new policy is being implemented for CS 306 starting Spring 2012:
You will be required to have submitted particular lab/homework assignments in order to be allowed to sit for each exam! If you fail to submit (suitable) versions of the required assignments, you will not be allowed to sit for the associated exam, and you will receive an automatic *zero* for that exam.
- This new policy is being instituted because significant fractions of CS 306 students have been failing to complete the programming assignments, in spite of this being critical to meeting course goals. It is also being tried to allow students to learn collaboratively, while minimizing the effects of “cheating.”
- *You must score at least 25% on an assignment for your submission to count toward exam admittance requirements!*
- See the course webpage for a list of the assignments that must be submitted to be allowed to sit for each of the exams.
- Under this new policy, your course grade will be based on *exam performance alone*. Each exam will consist of both *short answer and coding (programming) problems*. It is an important goal of this course that you learn to program in C, using system calls, so you will be required to demonstrate this skill by writing code on the exams. The coding problems will constitute about 50% of the total points on each exam.
- Even though the precise scores on assignments will not affect your final grade, assignments will still be graded so that you can see what you know and what you do not. (Scores on the assignments will be used to determine whether the assignment counts toward the requirements for admittance to an exam.)
- Assignments will *not* be accepted for scoring after the final due date/time has been set unless a student can document a significant reason why the assignment could not be submitted on time. (The only exception to this policy is that students will be granted a 24 hour “grace period” to submit an assignment in case of an undocumented “illness.”)

- Makeup exams will be offered only for *documented* conditions such as illness or university-related travel.
- The final exam will be comprehensive, but is *optional*. Students will be informed of what their course grade would be without the final exam, and if they are satisfied, they need not take the final exam (their grade will then be based on the average of their three in-class exam scores). Thus, the final exam serves as a chance for students that have had problems on one or more exams to demonstrate mastery of the material and potentially improve their course grade. Note, however, that the final exam score does not replace any of the three in-class exam scores, it merely averages with them. This will be the case even if a student received a zero on an exam due to failure to submit the assignments required for exam admission.
- Final course letter grades will be “curved” (e.g., rather than A’s being limited to score averages over 90%, an average of even 87% might result in an A). However, the only way to *guarantee*, say, an A, is to achieve an average exam score of better than 90%. Students must generally end up with an average of at least 60% to receive a grade of C or better in this course (which is what is required by both the BS and BA degrees in CS).

Lab (Programming) Assignments:

- While the lab assignments will not be a direct component of your grade, they play a critical role in you acquiring the skills you are to learn from the course—as well as those needed to do well on the exams. *It is simply impossible to learn to program well without spending significant time programming!* Virtually no students do well on the CS 306 exams unless they have invested enough time on the programming assignments.
- Because of the role that assignments play in allowing you to sit for exams, *it is critical to your success in this course that you begin working on each lab assignment as soon as possible after it is assigned.* One of the most difficult aspects of software development is estimating in advance how long it will take to complete a program (including debugging). Even professionals routinely *underestimate* the time that will be required—often by a significant amount. Do not simply guess that a lab will take you around four hours, then wait to start until 8pm the night before the lab is due. Not only can such a decision lead to you failing this class, the instructors have read enough emails sent by students at 4am the day a lab is due to know just how distressing it is to be discovering the “joys” of pointers and the like under deadline conditions.
- Lab (programming) assignments will be provided in relatively detailed *program specification* documents. You should treat these document as if they are being provided by your “boss” and your job depends on how well your programs adhere to the specs. Programming provides many avenues for creativity, but not when it comes to a program spec; you are free to meet the spec any way you want, but you must provide a program that is consistent with the spec. Far too many students lose points on their assignments because they fail to fully read the specification or because they fail to take it seriously. For example, if you are told to use a certain prompt or format output in a certain way, then if you choose to use a different prompt or format the output differently, your program is *incorrect* and points will be lost.

- *Programs that do not compile (contain syntax errors) or do not run through at least some required functionality, will not be graded!* We are not going to spend vast amounts of time trying to figure out how much of an assignment you completed by examining source code, when you were not willing to spend enough time to complete the assignment. Submissions that do not compile or do not run at least partially will receive an automatic zero (and so will not count toward admission to an exam).

Platforms and Facilities:

- Programming for the course will be done in C (not C++).
- The official OS platform for the course is *Linux* and the official C compiler is *GCC*.
- The CS Dept will provide you an account so you can access its Linux workstations, which are located in the *CS Dept computer lab in Faner 2102*. The PCs in this lab dual-boot Ubuntu Linux and Windows.
- While various Linux distributions are available free of charge, this course does *not* cover Linux installation/configuration (see CS 406), and the professor and TA *cannot provide support* for students doing such installs. Students are neither required nor expected to have a personal Linux machine available.
- While the CS Dept workstations currently run Ubuntu Linux, virtually all Linux distributions should be compatible and will have GCC and other development tools available (though these tools may not be installed by default). Other UNIX-like OS's (e.g., Mac OS X, a BSD, or Solaris) should also be usable, but are *not officially supported*. If you choose to use one of them instead of Linux, it is up to you to make certain your programs compile and run properly on the CS Dept. Linux workstations before you submit them.
- *Windows OS's are definitely not compatible and should not be used for development in this course.* Most of the course will involve Linux/UNIX *system programming*, so it will be impossible for you to compile or run your programs under any version of Windows OS. Basically all you could do under Windows is edit your C source files.
- You are strongly encouraged to do all your work for the course on Linux machines (either the CS Dept workstations or a personal Linux machine). This will definitely help you become familiar with the Linux commands that you will have to know for the exams, and will avoid compilation and other problems. Students that regularly use Linux systems for their coursework always perform better on the exams than those who choose to work via Windows machines.
- Be aware that the CS Dept lab is not open 24/7, so if you need to do your work on the CS Dept. workstations, be sure to plan your time accordingly.
- Two CS Dept. Linux workstations are *remotely accessible* via SSH: `pc00.cs.siu.edu` & `pc01.cs.siu.edu`. While this may allow you to work on assignments even when the workstation lab is closed, be aware that you are responsible for figuring out how to remotely access these machines and transfer files, and you will not be excused from getting assignments in on time if you encounter access problems (unless the machines cannot be accessed due to SIUC or CS Dept. problems).

- You will be instructed how to electronically submit your labs/programs for grading. Failure to follow the instructions may result in a score of zero for labs.

Collaboration and Cheating:

- Cheating on programming assignments is a significant problem throughout CS courses. The CS Dept has a page on *Academic Dishonesty* on its website, and this defines cheating in some detail. You should familiarize yourself with it.
- *If you cheat on an assignment in this class, you will receive a zero for the assignment.* This will in turn cause you to fail to meet the requirements for admission to one exam, which will severely impact your course grade. *A second violation will result in an immediate 'F' for the course.*
- While collaborating on assignments with classmates is considered cheating in many CS classes, the grading structure for this course has been changed in part to allow for *limited collaboration on the programming assignments. Collaboration on homework assignments is still not allowed!*
- **You will be allowed to work on lab (programming) assignments for CS 306 in teams of no more than three students.** If you choose to work in a team, every team member must confirm the team by:
 - (1) sending the instructor an email listing his fellow teammates prior to submission;
 - (2) including a *comment* at the top of submitted source file(s) listing all teammates.
- While some collaboration on lab assignments is being allowed, this does not necessarily mean that collaboration is the best way for you to work. If you are capable of completing the assignments by yourself, you will probably learn more and be better prepared for the exams than if you work in a team. On the other hand, developing code in conjunction with other students has the potential to be a useful learning experience. However, when students work “collaboratively” on a programming assignment, it is often the case that some team members will not learn everything they should, because they do not end up coding the entire assignment. If you work collaboratively, you must make certain that you understand every single line of the code that you submit. If you do not, you are likely to perform poorly on the exams.
- *Every line of code that you submit must have been written by you or by one of your teammates* (or have been provided by the instructor with the understanding that you could copy his code). The requirement for all members of a team to identify the team is to prevent “Bob” from copying code from “Bill” without his permission, because both Bob and Bill will have to say they worked as a team for it to be OK for Bob and Bill to share common code on their submissions.
- *You will be considered to have cheated on a lab assignments in this course if your submission contains code that originated from somebody other than you or one of your teammates!* In other words, incorporating code that somebody else wrote into your own solution is not allowed, whether that code came from a classmate (other than a teammate), a friend, the instructor, the TA, or the Internet. Starting with somebody else’s code is still cheating even if you slightly modify that code—such as by changing variable names, swapping the orders of a few statements, etc.

- If you don't understand why submitting somebody else's modified code is cheating, consider that this would be equivalent to taking a book somebody else had written, changing the names of the characters, and then claiming the "new" book as your own original work. Few people would consider that to be true.
- In fact, because computer software is inherently *copywritten* in the US and most of the rest of the world, copying somebody else's code without their permission is not just cheating, it is a *violation of law*. This is even the case for most "*free and open source software*," since virtually all such software requires that anyone using the code in any way must retain existing authorship and copyright notices in any derived code (which would obviously alert the instructor that you had cheated).
- Studies of cheating in CS classes have found that students drastically underestimate the ease with which instructors can identify copied code. No two independently developed programs of any length should look terribly similar since there will be a vast number of alternative ways to meet the program specifications. Students also seem to forget that the instructor/TAs can perform the very same Internet searches that they can, and so can find the same publicly available code. While it is fine to do Internet searches to get ideas about how to implement certain functionality in your programs, cutting and pasting somebody else's code into your program is cheating (even if you then modify it slightly).

Emergency Procedures:

Southern Illinois University Carbondale is committed to providing a safe and healthy environment for study and work. Because some health and safety circumstances are beyond our control, we ask that you become familiar with the SIUC Emergency Response Plan and Building Emergency Response Team (BERT) program. Emergency response information is available on posters in buildings on campus, available on BERT's website at www.bert.siu.edu, Department of Safety's website www.dps.siu.edu (disaster drop down) and in Emergency Response Guideline pamphlet. Know how to respond to each type of emergency.

Instructors will provide guidance and direction to students in the classroom in the event of an emergency affecting your location. It is important that you follow these instructions and stay with your instructor during an evacuation or sheltering emergency. The Building Emergency Response Team will provide assistance to your instructor in evacuating the building or sheltering within the facility.