Systems Programming Concepts

CS-202 **Winter 2010**

Course Information

Catalog Course Description: Fundamental system programming concepts are examined using the C programming language. Topics include: machine organization, data representation, interrupt handling, I/O, file management, dynamic structures, parameter passing, memory management, system calls, process creation, process control, interprocess communication, and language interfaces.

Prerequisites: CS-102, Computing and Algorithms II.

Course Topics:

- C programming: problem solving, program design, implementation, and testing.
- C programming constructs: variables, constants, literals, and comments.
- Programming style, white space for clarity of program.
- Data types.
- Program statements including assignment, invocation, control flow via decision, case, and loop.
- Operating system standardization and portability.
- Low level and high level file I/O including buffering.
- Atomic and nonatomic operations.
- File topics including file types, directories, permissions, ownership, access, and file systems.
- System data files.
- Unix process environment including memory layout and memory allocation.
- Process control including process creation, process termination, process execution, and interprocess communication.
- Signals, pipes, coprocesses, and FIFOs.

Course Objectives: By the end of this course, you should be able to demonstrate the ability to do all of the tasks listed below:

- Design, implement, compile, test, and run a C computer program which uses system calls.
- Write a program using appropriate documentation and style for effective communication with a human reader.
- Write a systems program which is portable to another POSIXcompliant operating system.

- Create, terminate, and execute processes via calls to system routines.
- Implement simple communication between processes.
- Write code to handle system interrupts.

Administrative Information

Required Textbooks:

- Molay, *Understanding Unix/Linux Programming*, Prentice Hall, 2003.
- Stevens & Rago, *Advanced Programming in the Unix Environment*, 2nd edition, Addison Wesley, 2008.

Optional Textbooks:

- King, *C Programming: A Modern Approach*, 2nd edition, W.W. Norton, 2008.
- Harbison & Steele Jr., *C: A Reference Manual*, 5th edition, Prentice Hall, 2002.

Instructor: Jim Huggins

Email: `jhuggins@kettering.edu`

Office: AB 2-300G

Office Hours: WF 10:00am-11:30am, *or by appointment*

Office Phone: 762-9500, x5439

Course WWW Page: See `http://blackboard.kettering.edu`

Course Overview

CS-202 will acquaint you with the world of systems programming: writing programs that interact directly with a modern operating system (in this case, Linux). You will learn about how Linux works from a service provider's perspective, learning the kinds of system calls that can be made by programs in order to take advantage of the services provided by the operating system. In the process, you will begin to learn about how modern computers operating systems are organized: a skill which will be used in successor courses like CE-320 (Microcomputers) and CS-451 (Operating Systems).

The material in this course is cumulative in nature; concepts learned in one week will depend on concepts learned in previous weeks. It is thus vital that you keep up with course concepts as the course proceeds; if you find yourself falling behind, *please* see the instructor as soon as possible.

Course Requirements and Grading

Exams

A midterm examination, worth 25% of the final course grade, is tentatively scheduled for ***12 February (5th Friday)***. A final examination, worth 25% of the final course grade, will be held during the final examination period. Both examinations will be closed-book, closed notes. Except for emergencies, makeup exams must be arranged *in advance* of the announced examination date. Naturally, examinations should be entirely your own work.

Examinations will cover material presented in lecture and announced sections of the textbooks. You are responsible for all material presented in class.

Homework

There will be several programming assignments, each varying in length and importance. Dates and relative weights for each assignment will be announced. Homework will be worth a cumulative one-half of the final course grade.

Each programming assignment will be graded on the basis of functionality (50%), design requirements and elegance (25%), and style (25%). A style sheet for programming assignments will be distributed separately.

Unless otherwise stated in the assignment, programming assignments should be *entirely your own work*. You are permitted to discuss homework assignments with fellow classmates in a general nature; you are *not* permitted to jointly develop algorithms and/or code. You are not permitted to copy algorithms and/or code from other students or from outside sources, or to give or loan code to other students. You are not permitted to use code-generating programs to assist you in writing your code.

Late programming assignments will be assessed a 20% penalty for the first day or portion thereof after the specified due date, and an additional 10% penalty per day or portion thereof afterwards. *All* programming assignments must be submitted for grading in order to pass the course; students who fail to do so may receive a grade of F for the course.

Course Grades and Policies

Final course grades will initially be computed using the relative weights described above. A class curve will be applied. Individual grades may be adjusted to reward improvement over the course of the semester. The official interpetations of the Kettering University Grading Scheme in the Kettering University Catalog will be used to guide the above adjustments.

Questions regarding individual scores on assignments and exams should be addressed to the instructor within one week of receiving the score.

All suspected cases of academic dishonesty will be handled in *strict* accordance with Kettering University policy. Any questions regarding appropriate behavior should be cleared with the instructor in advance.

Attendance

Attendance at all lectures and laboratory sessions is *strongly encouraged.* In particular, attendance at laboratory sessions is strongly encouraged; laboratory sessions are an opportunity to work on your current programming assignment in a dedicated, uninterrupted environment, with the assistance of the instructor.

You are permitted to attend any (or all!) scheduled laboratory sessions as long as empty seats are available; priority on empty seats will (naturally) be given to those registered for the section currently being held.

The following laboratory sessions are being held (all in 3-509 AB):

- Tuesday, 3:35pm-5:40pm (Section 4)
- Wednesday, 8:00am-10:00am (Section 2)
- Wednesday, 3:35pm-5:40pm (Section 3)