CS 4400: Computer Systems

Administrative Details and Syllabus
Fall 2008

Description. The objective of this course is to help students bridge the gap between high-level programming and actual computer systems: processors, the memory hierarchy, operating systems, compilers, linkers, assemblers, networks, and more.

Our basic goal is to understand how a computer works, so that as programmers we can make it work efficiently. Thus, this course is an introduction to computer systems from a programmer's point of view.

The prerequisites for this course are CS 3500 (Software Practice I) and CS 3810 (Computer Organization).

Instructor. D. Erin Parker. *Email:* `parker@cs.utah.edu`. *Office:* 3190 MEB *Office Hours:* See the class web page.

Teaching Assistants. TBA. See the class web page for the TA consulting schedule.

Class Meetings. Mondays and Wednesdays 1:25-2:45p in 101 WEB.

Communication. The class web page is `http://www.coe.utah.edu/~cs4400`. It will be updated throughout the semester with the lecture schedule, assignments, links to course handouts, and more.

For questions outside of class and office hours, students are encouraged to use email.

The course staff will use the class mailing list (`cs4400@eng.utah.edu`) to send urgent messages to everyone in the class, such as corrections to assignments or changes in due dates. Students are not able to send mail to the class list. Students *must* subscribe to the class mailing list by visiting `https://sympa.eng.utah.edu/sympa/info/cs4400`.

Students who would like to ask a question of the course staff should use the staff mailing list (`teach-cs4400@eng.utah.edu`). The instructor or a teaching assistant will respond to each question directly.

Text. The required course text is *Computer Systems: A Programmer's Perspective* by Randal E. Bryant and David R. O'Hallaron, 1st edition, Prentice-Hall, 2003 (ISBN: 0-13-034074-X), "CS:APP". A highly recommended book is *The C Programming Language* by Brian W. Kernighan and Dennis M. Ritchie, 2nd edition, Prentice-Hall, 1988 (ISBN: 978-0131103627), "K&R".

Lectures. This class will meet for lecture two times a week for one hour and twenty minutes. The instructor will make use of slides during lecture, and the slides covered during each lecture will be posted on the class web page following the lecture. Students are encouraged to take notes in class and should not expect to rely solely on posted slides to recall the material covered in each lecture.

Reading. The lecture schedule is posted on the class web page. Following each class meeting, the schedule will be updated to reflect the material actually covered by the lecture that day and

to indicate the reading assignment for the next lecture. By the end of the semester, the lecture schedule will be a record of all the material covered in this course.

Problem Sets. At each class meeting, a number of problems from the textbook will be assigned. Completing the these problem sets is for the student's own benefit and *not for credit*. Solutions to some or all of the problems will be discussed during the next class meeting, and a healthy amount of class participation is expected. If throughout the semester, participation ever fails to meet expectations, completion of problem sets may become *for credit*.

Lab Work. Because this is a 4-unit course, there is a significant amount of lab work in the form of programming assignments.

The lab work makes heavy use of C, Unix, and the Intel IA32 (x86) architecture. Students not currently fluent in any of these three topics should not panic, as this course will cover them in more detail throughout the semester. However, there is an assumption that students have some familiarity with C or C++. Students should be prepared to learn some of the C programming language on their own, for which the Kernighan and Ritchie reference text will be very useful.

To behave properly, all lab work must use an x86 processor (either AMD or Intel) that has a Unix OS. *Code must be in ANSI standard C—nothing else will work.* Grading of assignments will be done using CADE Lab 1 machines. Students who choose to develop their code on any other machine are strongly encouraged to run their programs on a CADE Lab 1 machine before turning it in. *There will be no credit for programs that do not compile and run on a CADE Lab 1 machine*, even if they run somewhere else.

CADE Lab 1 machines are numbered 1-48 and have names `lab1-x.eng.utah.edu`, where `x` is the machine number. For more information on the CADE lab and how to remotely log into these machines, see `http://www.cade.utah.edu`.

Late Policy. Assignments are due at 11:59p on the due date via electronic submission. Late assignments are accepted according to the following rules.

- Assignments are not accepted more than 3 days after the due date.
- Assignments submitted any time X days after the due date (midnight to 11:59p) are penalized $X * 10\%$ of the assignment grade.

Grading. The final course grade will be based on six evenly-weighted lab-work assignments (50% total), two evenly-weighted midterm examinations (30% total), and a final examination (20%).

The following scale is used to assign letter grades.

		89-87	B+	79-77	C+	69-67	D+		
100-93	A	86-83	B	76-73	C	66-63	D	59-0	E
92-90	A-	82-80	B-	72-70	C-	62-60	D-		

As indicated above, if problem set participation ever fails to meet expectations, a portion of the final course grade will be allotted to problem sets.

Students who wish to appeal a grade on an assignment or an exam, must do so within one week of receiving the grade.

The final exam for this class will be on Monday, December 15, from 1:00-3:00p in 101 WEB.

Working Together. Students are encouraged to discuss lab-work assignments with fellow classmates, but each student is responsible for writing her own answer. *Cheating is:* sharing code or other electronic files either by copying, retyping, looking at, or supplying a copy of a file. *Cheating is not:* discussing concepts, answering questions about concepts or clarifying ambiguities, helping someone understand how to use the computer systems or basic tools (compiler, debugger, etc.), or helping with high-level design issues or debugging.

Except for the last assignment, each assignment is to be done individually. For all assignments, the solution submitted by each assignment will be checked against the solutions of other students (from this year's class, as well as, previous years) for anomalies. If an anomaly is found that cannot be explained satisfactorily, the students involved will fail the course.

Of course, there must be no collaboration during examinations. Please see the University of Utah Student Code (`www.admin.utah.edu/ppmanual/8/8-10.html`) for a detailed description of the university policy on cheating.

Students with Disabilities. The University of Utah seeks to provide equal access to its programs, services, and activities for people with disabilities. Students who need accommodations in this class should give reasonable prior notice to the Center for Disability Services, 162 Olpin Union Building, 581-5020 (V/TDD). CDS will work with the student and instructor to make arrangements for accommodations.

Syllabus. The following are the key topics planned for study, the approximate number of lectures devoted to each, and the corresponding chapters in the course text.

Getting Started (1 lecture) Chapter 1
- Administrative details
- Overview of computer systems

Representing Information (8 lectures) Chapters 2-3
- Bits and bytes (information storage)
- Integers (representation, arithmetic)
- Floating point (representation)
- x86 machine-level code (accessing information, operations)
- Control flow (jumps, branches)
- Procedures (run-time stack, recursion)
- Data (arrays, pointers, structures, alignment)

Optimizing Code (2 lectures) Chapter 5
- Optimizing compilers
- Loops
- Branch prediction
- Memory performance

The Memory Hierarchy (2 lectures) Chapter 6
- Different kinds of memory
- The principle of locality
- Cache memory

Running Programs on a System (8 lectures) Chapters 7-10
- Linking
- Exceptions
- Processes
- Signals
- Virtual memory and address translation
- Dynamic memory allocation

Interaction Among Programs (3 lectures) Chapters 12-13
- Network programming
- Concurrent programming