Introduction to Computer Systems

Syllabus

Web Page

http://www.cs.northwestern.edu/~pdinda/ics-f02

Instructor

Peter A. Dinda
1890 Maple Avenue, Room 338
847-467-7859
pdinda@cs.northwestern.edu
Office hours: Thursdays, 2-4pm

Teaching assistant

Dong Lu
1890 Maple Avenue, Room 234
847-467-6947
donglu@cs.northwestern.edu
Office hours: Mondays, 3-5pm

Location and Time

1890 Maple Avenue, CS Department classroom, WF 11-12:20

Prerequisites

Required	CS 211 or equivalent
Required	Experience with C or C++
Recommended	CS 311 or equivalent

Textbook

Randal E. Bryant and David R. O'Hallaron*, Introduction to Computer Systems: A Programmer's Perspective,* Prentice Hall, 2003, (ISBN 0-13-034074-X) (Required - Textbook)

- Details on http://csapp.cs.cmu.edu

Brian W. Kernighan and Dennis M. Ritchie, *The C Programming Language, Second Edition*, Prentice Hall, 1988 (ISBN 0-131-10370-9) (Required)

- Definitive book on C

Richard Stevens, *Advanced Programming in the Unix Environment*, Addison-Wesley, 1992 (ISBN 0-201-56317-7) (Recommended)

- Describes how to think like a Unix systems programmer

Objectives, framework, philosophy, and caveats

This course has four purposes. First, you will learn about the hierarchy of abstractions and implementations that comprise a modern computer system. This will provide a conceptual framework that you can then flesh out with courses such as compilers, operating systems, networks, and others. The second purpose is to demystify the machine and the tools that we use to program it. This includes telling you the little details that students usually have to learn by osmosis. In combination, these two purposes will give you the background to understand many different computer systems. The third purpose is to bring you up to speed in doing systems programming in a low-level language in the Unix environment. The final purpose is to prepare you for upper-level courses in systems.

This is a learn-by-doing kind of class. You will write pieces of code, compile them, debug them, disassemble them, measure their performance, optimize them, etc.

This course is ideally taken after CS 211 early in your academic career.

Resources

You'll be able to do the programming assignments on any modern Linux machine. Consult the course web for how to get an account on the TLAB machines. It should be possible to do the assignments using the Cygwin environment on Windows, but they will be graded in a Linux environment, so be sure your code works on the TLAB machines.

Labs

There will be four programming labs. Their goal is to make you apply the concepts you've learned and to gain familiarity with Unix tools that can help you apply them. Labs should be done in groups of two.

Homework

Four problem sets will be assigned. Their goal is to help you improve your understanding of the material. Homework should be done alone.

Exams

There will be a midterm exam and a final exam. The final exam will not be cumulative.

Grading

10 %	Homeworks (2.5% per homework)
50 %	Programming labs (15/15/5/15% or 12.5/12.5/12.5/12.5% depending)
20 %	Midterm (covers first half of the course)
20 %	Final (covers second half of the course)

Final grades will be computed in the following way. A final score from 0 to 100 will be computed as a weighted sum of the homeworks, programming labs, and the exams. Scores greater than 90 or greater than 90th percentile will be assigned As, scores greater than 80 or greater than 80th percentile will be assigned Bs, scores greater than 70 or greater than 70th percentile will be assigned Cs, scores greater than 60 or greater than 60th percentile will be assigned Ds, and the remainder will be assigned Fs. Notice that this means that if everyone works hard and gets >90, everyone gets an A. Please choose wisely where you use your time.

Peter ultimately assigns all grades. If you have a problem with a grade, you are welcome to bring it up with either Peter or Dong, but only Peter is empowered to change grades.

Late Policy

For each calendar day after the due date for a homework or a lab, 10% is lost. After 1 day, the maximum score is 90%, after 2 days, 80%, etc, for a maximum of 10 days.

Cheating

Since cheaters are mostly hurting themselves, we do not have the time or energy to hunt them down. We much prefer that you act collegially and help each other to learn the material and to solve development problems than to have you live in fear of our wrath and not talk to each other. Nonetheless, if we detect blatant cheating, we will deal with the cheaters as per Northwestern guidelines.

Schedule

Lecture	Date	Topics	Readings	Homework/Labs
New student week				
1	9/25 W	Mechanics, Introduction, overview of abstractions using web request-response	Chapter 1	Data lab out
2	9/27 F	Physics, transistors, photolithography, Moore's Law, bits, bytes, and logic	2, 2.1, handout	
3	10/2 W	Integers and integer math	2.2-2.3	HW 1 out,
4	10/4 F	Floating point	2.4-2.5	

5	10/9 W	The Machine Model – instruction set architecture, microarchitecture, and basic instructions	3, 3.1-3.5, 5.7	HW 1 in, HW 2 out
6	10/11 F	Control flow	3.6	Data lab in Bomb lab out
7	10/16 W	Procedures	3.7	
8	10/18 F	Data	3.8-3.11	
9	10/23 W	Advanced machine code	3.12-3.16	
10	10/25 F	Memory and cache	6, 6.1-6.4	HW 2 in, HW 3 out
11	10/30 W	Cache performance	6.5-6.7	Bomb lab in,
12	11/1 F	Linking	Chapter 7	
Midterm exam, Monday 11/4, 6-8:30pm				
13	11/6 W	Exceptional control flow	8,8.1-8.4	Exploit lab out
14	11/8 F	Exceptional control flow	8.5-8.8	HW 3 in
15	11/13 W	Virtual memory	10, 10.1-10.6	
16	11/15 F	Memory system	10.7-10.8	Malloc lab out, Exploit lab in
17	11/20 W	Memory allocation	10.9-10.13	
18	11/22 W	Input and Output	Chapter 11	HW 4 out
19	11/27 F	Network programming	Chapter 12 handout	
Thanksgiving break				
20	12/4 W	Concurrency	Chapter 13 handout	
21	12/6 F	Distributed systems + wrap-up	handout	Malloc lab in HW 4 in
Finals week – Exam is Friday 12/13, 9-11am				

Note that in the latter part of the course, we will cover Chapters 11-13 at a very high level. I want you to read these chapters, but I will not cover them in their entirety in class.

We will skip Chapter 4 (Processor Architecture), 5 (Performance Optimization), and 9 (Measuring Execution Time). Chapter 4 is worth reading if you're interested in how a simple processor with an Intel-like instruction set is implemented. Chapter 5 is all about understanding how to make programs run faster. Chapter 9 is all about how to measure how fast programs run.

I may replace some of the material toward the later part of the course with material on performance optimization (i.e., Chapters 5 and 9) and replace the exploit lab with a performance optimization lab. I have not decided whether to do this or not and will keep you posted.