15-213: Introduction to Computer Systems

Randal E. Bryant and David A. Eckhardt

Computer Science
Carnegie Mellon University
Spring 2008

1 Organization

Instructors:

Randal E. Bryant
Randy.Bryant@cs.cmu.edu
NSH 4305, x8-8821
Fri., 3:00–4:00 (NSH Atrium)

David A. Eckhardt
de0u@andrew.cmu.edu
WeH 3503, x8-6720
TBA

TAs:

Nathaniel Bauernfeind
nbauernf@andrew.cmu.edu
Wed., 4:30–5:30, (WeH 3108)

Tessa Eng
yeng@andrew.cmu.edu
Sun., 4:00–5:00, (WeH cluster)

Austin McKinley
amckinle@andrew.cmu.edu
Tue., 3:00–4:00, (WeH 3108)

Pratyusa Manadhata
pratyus@cs.cmu.edu
Tue., 4:00–5:00, (WeH 3108)

Allison Naaktgeboren
pinkrobot@cmu.edu
Sun., 3:00–4:00, (WeH cluster)

Brett Simmers
bsimmers@andrew.cmu.edu
TBA

Lawrence Tan
kltan@andrew.cmu.edu
Wed., 3:30–4:30, (WeH 3108)

Owen Yamauchi
ody@andrew.cmu.edu
Wed., 5:30–6:30, (WeH 3108)

Please see the class Web page for up-to-date office hours.

Course Assistants:

Cindy Chemsak, NSH 4303, x8-7884, cindyc@cs.cmu.edu

Lecture:

Tuesday, Thursday 1:30–2:50pm, Doherty Hall 2315

Recitations:

A	Mon	10:30–11:20	DH 1211	Tessa
B	Mon	11:30–12:20	DH 1211	Owen
C	Mon	12:30–1:20	PH A21	Nate
D	Mon	1:30–2:20	BH 237B	Brett
E	Mon	2:30–3:20	PH A22	Austin
F	Mon	9:30–10:20	PH 226C	Lawrence
G	Mon	11:30–12:20	PH A21	Pratyusa
H	Mon	1:30–2:20	PH A21	Allie

Class Web Page: `http://www.cs.cmu.edu/~213`

Class Message Board: `http://autolab.cs.cmu.edu`
Note: This is the only message board your instructors will be monitoring. We will not be using the Andrew or Blackboard message boards for this class.

Contacting Teaching Staff: Send email to `15-213-staff@cs.cmu.edu`. This will go to all of the teaching staff.

2 Objectives

Our aim in 15-213 is to help you become a better programmer by teaching you the basic concepts underlying all computer systems. We want you to learn what really happens when your programs run, so that when things go wrong (as they always do) you will have the intellectual tools to solve the problem.

Why do you need to understand computer systems if you do all of your programming in high level languages? In most of computer science, we're pushed to make abstractions and stay within their frameworks. But, any abstraction ignores effects that can become critical. As an analogy, Newtonian mechanics ignores relativistic effects. The Newtonian abstraction is completely appropriate for bodies moving at less than $0.1c$, but higher speeds require working at a greater level of detail.

Oversimplifying matters somewhat, our $21x$ sequence works as follows: 211 is based on a simplified model of program execution. 212 builds further layers of abstraction. 213 introduces greater detail about system behavior and operation. This greater detail is needed for optimizing program performance, for working within the finite memory and word size constraints of computers, and for systems-level programming.

The following "realities" are some of the major areas where the abstractions we teach in 211/212 break down:

1. *Int's are not integers, Float's are not reals.* Our finite representations of numbers have significant limitations, and because of these limitations we sometimes have to think in terms of bit-level representations.

2. *You've got to know assembly language.* Even if you never write programs in assembly, The behavior of a program cannot be understood sometimes purely based on the abstraction of a high-level language. Further, understanding the effects of bugs requires familiarity with the machine-level model.

3. *Memory matters.* Computer memory is not unbounded. It must be allocated and managed. Memory referencing errors are especially pernicious. An erroneous updating of one object can cause a change in some logically unrelated object. Also, the combination of caching and virtual memory provides the functionality of a uniform unbounded address space, but not the performance.

4. *There is more to performance than asymptotic complexity.* Constant factors also matter. There are systematic ways to evaluate and improve program performance

5. *Computers do more than execute instructions.* They also need to get data in and out and they interact with other systems over networks.

By the end of the course you will understand these "realities" in some detail. As a result, you will be prepared to take any of the upper level systems classes at Carnegie Mellon (both CS and ECE). Even more important, you will have learned skills and knowledge that will help you throughout your career.

3 Textbook

The primary textbook for the course is

> Randal E. Bryant and David R. O'Hallaron, *Computer Systems: A Programmer's Perspective*, Prentice Hall, 2003.

In addition, we require you to have the following reference book on the C programming language:

> Brian W. Kernighan and Dennis M. Ritchie, *The C Programming Language, Second Edition*, Prentice Hall, 1988.

This is the classic *K & R* book, the standard against which all reference manuals are compared. It is an essential part of every computer scientist's library.

4 Course Organization

Your participation in the course will involve five forms of activity:

1. Attending the lectures.

2. Preparing for and participating in the recitations.

3. Laboratory assignments.

4. Reading the text.

5. Exams

Attendance will not be taken at the lectures or recitation sections. You will be considered responsible for all material presented at the lectures and recitations.

Lectures will cover higher-level concepts. Recitations will be more applied, covering important "how-to's", especially in using tools that will help you do the labs. In addition, the recitations will help clarify lecture topics and describe exam coverage.

The textbook contains both *practice problems* within the chapter text and *homework problems* at the end of each chapter. The intention is that you work on the practice problems right as you are reading the book. The answers to these problems are at the end of each chapter. Our experience has been that trying out the concepts on simple examples helps make the ideas more concrete. In addition, the schedule (at the end of this document and on the class web page) shows specific homework problems with each lecture topic. The intention is that you try these out and discuss them in the next recitation. You will find that you will get much more out of recitation if you have done some advance preparation.

The only graded assignments in this class will be a set of seven labs. Some of these are fairly short, requiring just one week, while others are more ambitious, requiring several weeks.

5 Getting Help

For urgent communication with the teaching staff, it is best to send electronic mail (preferred) or to phone.

If you want to talk to a staff member in person, remember that our posted office hours are merely nominal times when we guarantee that we will be in our offices. You are always welcome to visit us outside of office hours if you need help or want to talk about the course. However, we ask that you follow a few simple guidelines:

- Prof. Bryant is glad to meet with students, but he's also got a lot of other meetings. The best way to meet with him is to set up an appointment through his assistant, Cindy Chemsak (`cindyc@cs.cmu.edu`).
- Prof. Eckhardt normally work with his office door open and welcomes visits from students whenever his doors are open. However, if his door is closed, he is busy with a meeting or a phone call and should not be disturbed.
- Please send mail to arrange a meeting with your TA outside of office hours.

We will use the Web as the central repository for all information about the class. The class home page is at

```
http://www.cs.cmu.edu/~213
```

Using the Web, you can:

- Obtain copies of any handouts or assignments. This is especially useful if you miss class or you lose your copy.
- Find links to any electronic data you need for your assignments

- Read clarifications and changes made to any assignments, schedules, or policies.
- Post messages messages to make queries about the course, specific labs, or exams.

The lab assignments and class message board are offered through a Web service written by Prof. David O'Hallaron called *Autolab*. See the Autolab web page at `http://autolab.cs.cmu.edu` for more information.

6 Policies

Working Alone on Assignments

You will work on all assignments by yourself, unless notified otherwise in the assighnment description.

Handing in Assignments

All assignments are due at 11:59pm (one minute before midnight) on the specified due date. All handins are electronic using the Autolab system.

Handing in Late Assignments

Each student will receive a budget of five *grace days* for the course. Here is how grace days work:

- If you hand in an assignment k days late, then you receive full credit for the lab, but you will have spent k of your grace days. For example, if an assignment is due at 11:59pm on Thursday and you hand it in at noon on Saturday, then you will have spent 2 grace days. If you hand it in at 9am on Friday, then you will have spent 1 grace day.
- When you are out of grace days, you can no longer hand in assignments, and must make special arrangments with your professor, as described in the "Making up Exams and Assignments" section.
- Regardless of the number of grace days you have remaining, handins will not be accepted after the *end date* of the lab, which is typically 2 days after the due date.

Grace days are a tool to help you manage your time and to help smooth out burstiness in assignment due dates. We recommend that you conserve your grace days, saving them for the end of the term when things get most hectic.

Making up Exams and Assignments

Missed exams and assignments more than 2 days late can be made up on a case by case basis, but only if you make prior arrangements with Prof. Eckhardt. However you should have a good reason for doing so. It is your responsibility to get your assignments done on time. Be sure to work far enough in advance to avoid unexpected problems, such as illness, unreliable or overloaded computer systems, etc.

Appealing Grades

After each exam and lab assignment is graded, Prof. Eckhardt will send each of you a personalized email with your grade (as well as all of your previous grades). You have seven calendar days from the date he sends the email to appeal your grade.

Each TA in 15-213 has the authority to unilaterally change your lab grade, without permission from the professors. So if you have questions about the grade you received on a lab assignment, please talk first to the person in charge of the assignment, who will be clearly identified in the writeup.

If you are still not satisfied, please come and visit Prof. Eckhardt. If you have questions about an exam grade, please visit Prof. Eckhardt directly.

Final Grade Assignment

Each student will receive a numeric score for the course, based on a weighted average of the following:

- **Assignments:** The assignments will count a combined total of 60% of your score. The exact weighting of the different assignments will be determined near the end of the course based on our perception of the relative effort required. In any case, each lab will count 6–12% of your score. Since small differences in scores can make the difference between two letter grades, you'll want to make a serious effort on each assignment.
- **Exams:** There will be two in-class exams, each counting 10%, plus a final counting 20%.

Grades for the course will be determined by a method that combines both curving and absolute standards. The total score will be plotted as a histogram. Cutoff points are determined by examining the quality of work by students on the borderlines. Individual cases, especially those near the cutoff points may be adjusted upward or downward based on factors such as attendance, class participation, improvement throughout the course, final exam performance, and special circumstances.

Cheating

Each lab assignment must be the sole work of the student turning it in. Assignments will be closely monitored by automatic cheat checkers, and students may be asked to explain any suspicious similarities. The following are guidelines on what collaboration is authorized and what is not:

What is Cheating?

- *Sharing code or other electronic files*: either by copying, retyping, looking at, or supplying a copy of a file.
- *Sharing written assignments*: Looking at, copying, or supplying an assignment.
- *Using other's code.* Using code from previous offerings of 15-213, as well as from courses at other institutions.

What is NOT Cheating?

- Clarifying ambiguities or vague points in class handouts or textbooks.
- Helping others use the computer systems, networks, compilers, debuggers, profilers, or other system facilities.
- Helping others with high-level design issues.
- Helping others with high-level (not code-based) debugging..
- Using code from the CS:APP website or from the course web pages.

Be sure to store your work in protected directories.

The usual penalty for cheating is to be removed from the course with a failing grade. We also place a record of the incident in the student's permanent record.

7 Facilities: Intel Computer Systems Cluster

Intel has generously donated a cluster of 15 Linux-based 64-bit Xeon servers, specifically for 15-213, that we will use for all labs and assignments. The class Web page has details.

8 Class Schedule

Figure 1 shows the tentative schedule for the class. The reading assignments are all from the CS:APP book. The schedule also indicates suggested homework problems, the lab activities, and the lecturer for each class.

Any changes will be announced on the class message board. An updated schedule will be maintained on the class Web page.

Class	Date	Day	Topic	Reading	Problems	Labs	Lecturer
1	01/15	Tue	Overview	1			Both
2	01/17	Thu	Bits, Bytes, and Integers	2.1–2.3	2.44, 2.45, 2.49, 2.54		REB
3	01/22	Tue	Floating Point	2.4–2.5	2.59, 2.60, 2.61		REB
4	01/24	Thu	Machine Prog I - Overview	3.1–3.5	3.31		DAE
5	01/29	Tue	Machine Prog II - Control	3.6	3.34	L1 Due, L2 Out	REB
6	01/31	Thu	Machine Prog III- Procedures	3.7			REB
7	02/05	Tue	Machine Prog IV- Data	3.8–3.11	3.36		DAE
8	02/07	Thu	Mach. Prog V- Advanced	3.12–3.13, 3.16	3.24		REB
9	02/12	Tue	Program Optimization I	5.1–5.6, 5.14–5.15	5.3	L2 Due, L3 Out	REB
10	02/14	Thu	Program Optimization II	5.7–5.13	5.6		REB
11	02/19	Tue	Memory Hierarchy	6.1–6.3	6.2, 6.3, 6.4	L3 Due, L4 Out	DAE
12	02/21	Thu	Exam 1				n/a
13	02/26	Tue	Cache Memories	6.4	6.9-6.17		DAE
14	02/28	Thu	Linking	7	7.2, 7.3		DAE
15	03/04	Tue	Except. Control Flow I	8.1–8.4	8.1, 8.2, 8.3	L4 Due, L5 Out	DAE
16	03/06	Thu	Except. Control Flow II	8.5–8.8	8.19		DAE
	03/11	Tue	Spring Break				n/a
	03/13	Thu	Spring Break				n/a
17	03/18	Tue	Virtual Memory	10.1–10.6	10.4	L5 Due, L6 out	DAE
18	03/20	Thu	P6/Linux Memory System	10.7–10.8	10.14		DAE
19	03/25	Tue	Dynamic Storage Alloc I	10.9	10.6, 10.7		DAE
20	03/27	Thu	Dynamic Storage Alloc II	10.10–10.13	10.18		REB
21	04/01	Tue	System-level I/O	11	11.2, 11.3		REB
22	04/03	Thu	Exam 2				n/a
23	04/08	Tue	Network Programming	12.4	12.5	L6 Due, L7 Out	DAE
25	04/15	Tue	Web Services	12.5–12.7			REB
	04/17	Thu	Carnival				n/a
26	04/22	Tue	Guest Lecture				guest
28	04/29	Tue	Synchronization	13.5–13.8	13.7, 13.9, 13.10		REB
29	05/01	Thu	Course review			L7 due	DAE

Figure 1: 15-213 Class Schedule