CS 213: Introduction to Computer Systems

Bruce Maggs and Seth Copen Goldstein
Carnegie Mellon University
Spring 2003

1 Organization

Instructors:

Bruce Maggs	Seth Copen Goldstein
WeH 4123	WeH 7122
x8-7654	x8-3828
bmm@cs.cmu.edu	seth@cs.cmu.edu
Fri 2–3pm	Wed 11am–12pm

TAs:

Dave Koes	Jiin Joo Ong	Shaheen Gandhi	Mike Nollen	Greg Reshko
Section A	Section B	Section C	Section D	Section E
WeH 3723	WeH 3108	WeH 3108	WeH 3108	WeH 3108
dkoes@cs	jiinjoo@andrew	sgandhi@andrew	mnollen@andrew	reshko@cs
Tue 5–6pm	Tue 8–9pm	Fri 12:30–1:30pm	Mon 3–4pm	Wed 2–3pm

Class Assistant:
Dorothy Zaborowski
Wean 4116
(412) 268-3779
daz@cs.cmu.edu

Lecture:
Tue/Thu 9:00–10:20, Wean Hall 7500

Recitations:

A	Mon	10:30–11:20	OSC (Old Student Center) 203	Dave Koes
B	Mon	11:30–12:20	OSC (Old Student Center) 203	Jiin Joo Ong
C	Mon	12:30–1:20	OSC (Old Student Center) 203	Shaheen Gandhi
D	Mon	1:30–2:20	OSC (Old Student Center) 203	Mike Nollen
E	Mon	2:30–3:20	OSC (Old Student Center) 203	Greg Reshko

Web page: `www.cs.cmu.edu/afs/cs/academic/class/15213-s03/www/`

Newsgroup: `cmu.cs.class.cs213`
Staff Mailing List: `staff-213@cs.cmu.edu`

2 Objectives

Our aim in CS 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 recommend that you acquire a reference book on the C programming language. The following two are excellent choices

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

> Samuel P. Harbison and Guy L. Steele Jr., *C: A Reference Manual, Fifth Edition*, Prentice Hall, 2002.

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 7 labs. Some of these are fairly short, requiring just one week, while others are more ambitious, requiring several weeks. Most labs will be handed out in class on Thursday and due either one or two weeks later on a Wednesday.

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 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. Goldstein and Prof. Maggs normally work with their office doors open and welcome visits from students whenever their doors are open. However, if their doors are closed, they are busy with a meeting or a phone call and should not be disturbed.
- The TAs share offices with other students. To avoid disturbing these students, please send mail or zephyr before visiting a TA outside of office hours so they can arrange to meet you.

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

`www.cs.cmu.edu/afs/cs/academic/class/15213-s03/www/`

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.
- Read clarifications and changes made to any assignments, schedules, or policies.
- Find links to any electronic data you need for your assignments

We have also set up a news group for this class, `cmu.cs.class.cs213`. This group will be used by members of the teaching staff to post announcements and clarifications. You may also post to this group to make queries.

6 Policies

Working Alone on Assignments

Unlike previous terms, you will work on the first three laboratory assignments by yourself. However, you will work on the last four assignments with a partner.

Handing in Assignments

All assignments are due at 11:59pm (one minute before midnight) on the specified due date. All handins are electronic, usually consisting of one or more files that are to be copied to a specified directory. The writeup for an assignment will provide details about the handin procedure for that assignment.

Personal Late Days

You are given five (5) *late days*. This means that, over the course of the semester you may accrue up to five days of tardiness on lab submissions. You can turn one assignment in five days late, five assignments one day late, turn them all in on time, turn two in three days late and one in one day late, etc. It is completely up to you. Please note, the ue of any portion of a late day constitutes the use of a whole late date.

There is one very important exception. Unfortunately, you cannot use more than two late days on the last assignment unless there are special circumstances. This is for our mutual benefit. It allows you to focus on your exams during the exam period and it gives us time to get everything graded.

These days are *not necessarily procrastination days*. They are analagous to unspecified *personal days* in the work place. They cover personal needs, civic concerns such jury duty, vacation time, minor illnesses and injuries, family concerns, etc.

Please also take comfort in knowning that 15-213 comes with a complete benefits package. Akin to disability coverage in the workforce, we will, of course, work with you in the event of a major, usually unexpected disruptive event. Under these circumstances, we usually coordinate with your advisor or Assoc. Dean, or the office of Student Affairs. The important think to remember is that, in the event of something major, you should speak with one of us or another University as soon as possible. We will work together to help you through the circumstances in a reasonable way.

A couple of cautions related to late days:

- All members of a team must have enough remaining late days to submit an assignment late.
- Because the assignments are on the heels of each other, using late days can be a dangerous prospect. It might be very hard or impossible to recover from using more than one or two on an assignment. Please be careful. The risk is that an assignment submitted late might still be incomplete – while taking time away from the ”fresh slate” of a new assignment. Sometimes it is better to get a lower grade on one assignment and use the time to get started early on the next one.

Making up Exams and Assignments

Missed exams and assignments can be made up, but only if you make prior arrangements with Prof. Maggs. 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, homework, and assignment is graded, we 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 we send this email to appeal your grade.

If you have questions about the grade you received on an assignment (homework or lab), 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. Maggs. If you have questions about an exam grade, please visit Prof. Maggs 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 or partners 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.
- *Exploiting outside sources*: Using solutions or tools that automatically generate solutions from sources other than the course book, lectures, or staff.

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 debug their code.

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 25 Linux-based Pentium III servers, specifically for CS 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 in the new book. You can check on the class web page for suggested readings in K&R. The schedule also indicates suggested homework problems, the lab activities, and the lecturer for each class.

Any changes will be announced on the class news group. An updated schedule will be maintained on the class web page.

Class	Class	Topic	Reading	Problems	Labs	Lecturer
1	01/14	Overview	1		L1 Out	TBD
2	01/16	Bits and Bytes	2.1	2.44, 2.45		TBD
3	01/21	Integers	2.2–2.3	2.49, 2.54	L2 Out	TBD
4	01/23	Floating Point	2.4–2.5	2.59, 2.60, 2.61	L1 Due Friday	TBD
5	01/28	Machine Prog I - Overview	3.1–3.5	3.31		TBD
6	01/3	Machine Prog II - Control	3.6	3.34		TBD
7	02/04	Machine Prog III- Procs	3.7		L3 Out, L2 Due Wednesday	TBD
8	02/06	Machine Prog IV- Data	3.8–3.11	3.36		TBD
9	02/11	Mach. Prog V - Advanced	3.12–3.13, 3.16	3.24		TBD
1	02/13	Program Optimization I	5.1–5.6			TBD
11	02/18	Program Optimization II	5.7–5.16	5.14, 5.17, 5.18	L4 Out, L3 Due Wednesday	TBD
12	02/2	Memory Hierarchy	6.1–6.4	6.21, 6.23, 6.24		TBD
13	02/25	Cache Memories	6.5-6.8	6.25, 6.26, 6.27		TBD
14	02/27	Exam 1	Time & Location TBA			TBD
15	03/04	Linking	7	7.8, 7.12	L5 Out, L4 Due Wednesday	TBD
16	03/06	Except. Control Flow I	8.1–8.4	8.10, 8.11		TBD
17	03/11	Except. Control Flow II	8.5–8.8	8.19		TBD
18	03/13	Time Measurement	9	9.10, 9.11		TBD
19	03/18	Virtual Memory	10.1–10.6	10.11, 10.12, 10.13	L6 Out, L5 Due Wednesday	TBD
2	03/2	P6/Linux Memory System	10.7–10.8	10.14		TBD
21	04/01	Dynamic Storage Alloc I	10.9	10.15, 10.16		TBD
22	04/03	Dynamic Storage Alloc II	10.10–10.13	10.18		TBD
23	04/08	System-level I/O	11	11.7		TBD
24	04/1	Exam 2	Time & Location TBA			TBD
25	04/15	Internetworking	12.1–12.3		L7 Out, L6 Due Wednesday	TBD
26	04/17	Network Programming	12.4			TBD
27	04/22	Web Services	12.5–12.7	12.1		TBD
28	04/24	Concurrent Servers	12.8			TBD
29	04/29	Threaded Programming	13.1–13.3	13.22, 13.23	L7 Due Wednesday	TBD

Figure 1: CS 213 Class Schedule