CS 1109: Fundamental Programming Concepts

Summer 2011

This is an annotated version of the course syllabus that I designed for CS 1109 in the summer of 2011. Annotations in blue describe the rationale behind course design decisions that were made prior to teaching the class. Annotations in red are reflections based on the teaching experience.

Course Staff

Instructor
Raghuram Ramanujan
Upson 4143
raghu@cs.cornell.edu
Office Hours: 2:00-3:00PM, Monday through Thursday in Upson 328, or by appointment

Teaching Assistant
Chau Nguyen
Upson 4126
cnguyen@cs.cornell.edu
Office Hours: 2:00-3:00PM, Friday in Upson B7

In the future, maybe a week or so into class (once student class schedules have been finalized), take a straw poll to see if there are serious conflicts between scheduled office hours and other courses students are taking. In this case, due to the summer scheduling, office hours were in conflict with an intro to engineering course that the vast majority of students were enrolled in — talking to some students during lab uncovered this problem, and the situation was rectified by moving the hours to a more accessible time-slot. Also, don’t expect students to take the initiative to schedule alternative office hour appointments if they cannot attend your regular hours (maybe they don’t want to “inconvenience” the instructor?) — so offering additional help “by appointment” does not excuse poor scheduling.
Course Information

Website: http://www.cs.cornell.edu/courses/cs1109

Lectures:
- Section 1: Monday, Wednesday 10:00AM-11:00AM, Upson 111
- Section 2: Monday, Wednesday 11:30AM-12:30PM, Upson 215

Labs:
- Section 1: Tuesday, Thursday 10:00AM-11:00AM, Upson B7
- Section 2: Tuesday, Thursday 11:30AM-12:30PM, Upson B7

For future iterations of the course, when making room reservations, book the lab for every weekly scheduled class meeting, and not just the scheduled lab meetings. This allows more flexibility when crafting lesson plans; if you feel that a certain class would be better if taught in a lab, then you can take that decision and not worry about last-minute room availability issues.

About the Course

This course is designed to give you a solid grounding in the fundamentals of computer programming. You will cultivate your computational thinking and problem-solving skills, while simultaneously learning to design and code programs in the high-level programming language Matlab. Building upon fundamental ideas such as variables and operators, you will explore concepts such as arrays, control-flow structures, loops and file handling. You will also learn about Matlab’s data visualization tools and gain an appreciation for it’s usefulness as a scientific problem-solving tool. There are no prerequisites for the course aside from an understanding of basic high school algebra.

Course Goals and Objectives

Goals
- Appreciate the value of reduction and abstraction in problem-solving, and the power of computer programs as problem-solving tools
• Understand fundamental programming concepts and constructs such as variables, operators, conditional statements, control-flow modifiers, and loops.

• Understand good programming practices and learn to recognize and create high quality code that is efficient, elegant and readable.

This was my first attempt at backward design, though my process of defining course goals and objectives is more accurately described as a series of iterations of backward and forward design / refinement. I started with the basic question: What do I expect my students to learn from this course? This helped draft the initial set of goals and objectives. I then looked at sample syllabi from past iterations of the course to check if there were any glaring omissions. Talking to David Gries, who teaches the Java version of this course helped to define the scope of the course more sharply.

Specific Learning Objectives

The overarching objective of this class is to equip you with the skills necessary to transform a given problem specification (in English) into smaller, step-by-step instructions suitable for automatic execution. While you will gain proficiency in using Matlab, the skills you learn will be transferable to any other computational problem-solving task. More specifically, by the end of the course, you will be able to:

How successful was the course in meeting the articulated objectives? In the end-of-term course evaluations, students were asked to self-assess on a scale of 1–5 whether they felt they could carry out each of the tasks listed below (1 = strongly disagree that they can meet the objective, 2 = disagree, 3 = neutral, 4 = agree, 5 = strongly agree). The average score for each objective is listed below.

1. describe what computer programming and algorithmic thinking are 4.37/5.00

2. translate an English specification of a computational problem into pseudo-code 4.30/5.00
3. describe and use fundamental programming concepts such as variables and operators 4.90/5.00

4. describe and use programming constructs such as conditional expressions, if-then statements, loop structures and functions 4.87/5.00

5. create visualizations of data using Matlab’s plotting routines 4.47/5.00

6. write moderately complex Matlab programs that combine some or all of the above concepts to solve a well-specified problem 4.13/5.00

7. critique code based on style and efficiency and use good programming practices 4.43/5.00

The most glaring number here is the rating of 4.13 for objective (6). For most students, this was their first ever class of computer programming. Practice, in writing/debugging/testing code, is possibly the most important contributor to one’s development as a programmer. So given the compressed, 6-week timeline of the course, perhaps it is somewhat understandable that people aren’t completely confident in their ability to write moderately complex code. What can be done to give students more practice? Consider providing optional practice problems that students may choose to work on outside class. Also, perhaps optional recitations where we work through programming tasks, and I can walk students through my thought-process as I solve problems. Students should be rewarded for attending these sessions, so there is additional extrinsic motivation. The problem, of course, is that this adds more to student workloads in what is already a pretty heavy 2-credit course.

Also, for future iterations of the course, include a module on testing / debugging methods. Rather than discussing it informally with students on a one-to-one basis during labs / office hours as the need arises, reinforce the ideas using a dedicated lab session.

Course Structure and Expectations

1. Class meetings: The class meets four times a week for the duration of the summer session, with two lectures and two labs per week.
(a) **Lectures:** Lecture time will be dedicated to presentation and discussion of new concepts, and include individual and group problem-solving exercises.

(b) **Labs:** These will consist of programming activities that you complete together with a partner. The activities are designed to be completed within an hour’s time; if you are unable to complete a lab within the allotted class time, you must complete it in your own time and submit it by the end of the same day.

Please note that this course occurs over a compressed 6-week time period and as such will proceed at a faster pace than what you may be accustomed to. Therefore, it is essential that you not miss any classes to avoid falling behind. If you have to miss a lecture / lab meeting, it will be your responsibility to find out what you missed.

*The lecture + lab model is pretty standard for most courses of this nature. I did toy with idea of making every course meeting a lab meeting, so students could assimilate new concepts by trying them out right away. But I was also worried about propagating the myth that CS is all about writing programs (not to mention the potential for distractions with every student seated in front of a terminal).*

2. **Course Text:** There is no required course text for this class. However, there is a recommended text — *Getting Started with MATLAB 7: A Quick Introduction for Scientists and Engineers*, by Rudra Pratap. Several copies of the book have been placed on reserve at Olin Library for your benefit. Lesson summaries and PowerPoint slides will be available for download from the course website. Also, the course website links to a few Matlab tutorials that may serve as useful complementary reference material.

*I chose not to use a textbook since there are many excellent, free Matlab tutorials on the web to serve as reference and students would appreciate not having to shell out $100 on a book they may never use again.*

*One student commented about how having the class synched with a textbook would have been useful. Links to several online Matlab tutorials were provided on the course webpage; it’s unclear how many students used them. Would the course have benefited from closely following a*
textbook, as opposed to my own notes? Hard to tell — the lack of a reference text didn’t stop students from taking the initiative to learn about more advanced features of Matlab on their own via the web. But in the future, include explicit links to relevant sections of the tutorials after each class to make this discovery process for motivated students easier.

3. Quizzes: There will be a weekly 10-15 minute quiz during each of our Thursday class meetings. The primary purpose of these is to test your understanding of concepts that were covered the same week in class.

The quizzes were valuable more for the insights they provided into student learning, than as assessments. Average class performance on one quiz led to a revision of the course schedule to spend more time teaching loops. On another occasion, one student’s quiz performance prompted me to schedule a one-on-one tutoring session with him.

4. Homewones: There will be 6 – 7 homework assignments distributed over the course of the summer. Unlike the labs, you are expected to solve these by yourself. While you are allowed (and indeed, encouraged) to talk through your ideas with your classmates when solving the assigned problems, the final submission must be entirely your own work. Homework assignment and submission will be handled via the Computer Science department’s Course Management System (CMS):

http://cms.csuglab.cornell.edu

Given the tight schedule, I will not accept late submissions, unless there are extenuating circumstances. If you need an extension on a homework deadline, you must contact me ahead of time to make alternate arrangements.

The summer schedule ended up allowing for only 6 homework assignments, but they were well received. Students commented on the “interesting”, “creative” homeworks, and the “excitement of completing working code”. However, by general consensus, the homeworks were also deemed difficult (2.25/5.00). I think this tension is unavoidable. Interesting problems are inherently difficult! Articulating this sentiment to the class was important, for it made them more accepting of challenging assignments.
5. **Final Exam:** This course has a 2-hour final exam that will be held in a computer lab. It will consist of both written and programming components. The scope of the exam will be comprehensive, i.e., you can expect to be tested on any of the topics / concepts that we discuss during the duration of the course. The exam will hold no demons for you as long you are diligent and stay on top of the coursework throughout the summer.

There was some amount of flip-flopping with regards to whether an examination was truly an effective assessment method for this course. However, my conversation with David Gries convinced me that there is a utility to exams (and particularly lab exams, if feasible) — that they can expose misconceptions that are hard to detect via other means like programming assignments. His point was that you could still produce mediocre, but functional, software with a half-baked understanding of the underlying principles. This is not something that is easily identified via just programming assignments. To make the assessment more “authentic”, the test would be open-notes, open-browser. Students were permitted to refer to any online (or printed) resource, so long as it wasn’t another person. In real-life, you seldom have to code from memory, so it seemed unfair to deny students outside resources. Of course, in real-life, you can also just talk to another person, but if we are assessing individual learning, then the line has to be drawn somewhere.

6. **Anonymous Feedback:** At any point in the summer, you may submit comments about the course by following the “Feedback” link on the course homepage. All comments you submit will be transmitted to me in an anonymous fashion. You can use this service to comment about issues such as course pacing, clarity, teaching methodology etc.

No one made use of this form all summer, even though students were regularly reminded about the facility in class. It was far more effective to simply ask the students specific questions in class to get prompt feedback (for example: are office hours useful and accessible? how comfortable are student with certain content? etc.) and make the necessary adjustments. That said, there is still value to offering an anonymous feedback channel for students not comfortable raising certain issues in person (for example, academic integrity concerns etc.); just don’t expect students to go crazy with it.
Assessment and Grading

Your grade in this course will be determined on the basis of your performance in the weekly quizzes, labs, homework assignments and final exam. This class is graded on a pass / fail scale. Receiving a passing grade in this course requires demonstrating mastery of the material. To accomplish this, you must fulfil all of the following requirements:

- Score an average of at least 70% on the weekly quizzes
- Complete and submit at least 9 lab assignments in a timely fashion
- Score an average of at least 70% on the homework assignments
- Score at least 70% on the final exam

You can follow your progress in the class via CMS — once graded, assignment solutions and scores will be posted there.

The only thing to reconsider here is the 70% threshold — the bar was set too low, for some students skipped the last homework assignment altogether and still comfortably passed.

Academic Integrity

You are expected to abide by the Cornell University Code of Academic Integrity. A detailed description of the code may be found at the following URL:

http://cuinfo.cornell.edu/Academic/AIC.html

In short, I expect that any work you submit for this course will be your own. Collaboration is only permitted on lab assignments. You are highly encouraged to discuss material from lectures and ideas for tackling homework assignments with your classmates and course staff. However, any solutions you write (or type) up must be your own; at no point should you be in possession of any part of another student’s work (in electronic or written form).

If I establish that cheating has occurred, then all students involved in the case will receive a score of 0 for the assignment in question. Depending on the
severity of the violation, the punishment may include a failing grade for the course and disciplinary action from the University. *If you are in doubt regarding the bounds of acceptable collaboration, please contact me for clarification.*

**For Students with Disabilities**

In accordance with Cornell University policies, I am happy to make accommodations to meet the needs of students with disabilities. If you have a condition which you feel may have an adverse effect on your performance in the course, please contact me as soon as possible to discuss alternate arrangements.
### Class Schedule

The following is a preliminary plan for the progression of topics in this course. A more up-to-date schedule is available on the course webpage.

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Class</th>
<th>Assigned</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>06/27</td>
<td>Lec 1: Intro. to Computation, Syllabus</td>
<td>HW 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>06/28</td>
<td>Lab 1: Introduction to Matlab</td>
<td></td>
<td>HW 0</td>
</tr>
<tr>
<td></td>
<td>06/29</td>
<td>Lec 2: Relational &amp; Logical Operators</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>06/30</td>
<td>Lab 2: Input &amp; Output, Basic scripting</td>
<td>HW 1</td>
<td>Quiz 1</td>
</tr>
<tr>
<td></td>
<td>07/01</td>
<td>— FRIDAY —</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>07/04</td>
<td>Independence Day, no class</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/05</td>
<td>Lab 3: Vectors</td>
<td>HW 2</td>
<td>HW 1</td>
</tr>
<tr>
<td></td>
<td>07/06</td>
<td>Lec 3: The if statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/07</td>
<td>Lab 4: Pseudocode, algorithms</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/08</td>
<td>— FRIDAY —</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>07/11</td>
<td>Lec 4: The for-loop, colon notation</td>
<td>HW 3</td>
<td>HW 2</td>
</tr>
<tr>
<td></td>
<td>07/12</td>
<td>Lab 5: for-loops continued</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/13</td>
<td>Lec 5: The while-loop</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/14</td>
<td>Lab 6: Matrices</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/15</td>
<td>— FRIDAY —</td>
<td>HW 4</td>
<td>HW 3</td>
</tr>
<tr>
<td>4</td>
<td>07/18</td>
<td>Lec 6: Nested loops</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/19</td>
<td>Lab 7: Review of loops</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/20</td>
<td>Lec 7: Introduction to functions</td>
<td>HW 5</td>
<td>HW 4</td>
</tr>
<tr>
<td></td>
<td>07/21</td>
<td>Lab 8: More functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/22</td>
<td>— FRIDAY —</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>07/25</td>
<td>Lec 8: Data visualization</td>
<td>HW 6</td>
<td>HW 5</td>
</tr>
<tr>
<td></td>
<td>07/26</td>
<td>Lab 9: Data visualization</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/27</td>
<td>Lec 9: Intro to File I/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/28</td>
<td>Lab 10: More File I/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>07/29</td>
<td>— FRIDAY —</td>
<td>HW 7</td>
<td>HW 6</td>
</tr>
<tr>
<td>6</td>
<td>08/01</td>
<td>Lec 10: Selected Advanced Topics</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>08/02</td>
<td>Lab 11: Selected Advanced Topics</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>08/03</td>
<td>Lec 11: Review</td>
<td></td>
<td>HW 7</td>
</tr>
<tr>
<td></td>
<td>08/04</td>
<td>Lab 12: Review</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>08/05</td>
<td>— FRIDAY —</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>08/08</td>
<td>FINAL EXAM</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>