Course Information — ECS 120 — Spring 2010

You are responsible for everything on this handout — better read it!

Course homepage. www.cs.ucdavis.edu/∼rogaway/classes/120/spring10. Please visit the page regularly. It's a single click from my homepage, www.cs.ucdavis.edu/∼rogaway/.

Lectures. MWF 10:00–10:50 in 2016 Haring

Discussion sections. M 2:10–3:00 in 115 Hutchison

Discussion sections *do* begin the first day of class.

Instructor. Prof. Phillip Rogaway, Kemper #3009, rogaway@cs.ucdavis.edu, See the instructor's home page for office hours.

TA. Anhad Singh. e-mail: anhsing@ucdavis.edu Office hours (time and location) on the course home page.

Grading. The expected breakdown is as follows:

— **Problem sets**: **15%**
— **Quizzes**: **20%**
— **Midterm**: **20%**
— **Final**: **45%**

Caveats: (1) you cannot get a passing grade in the class without getting a passing grade on the final; (2) In assigning grades I may deviate from the stated numerical percentages if I see a compelling reason to do so.

There will probably be two quizzes and nine problem sets. If we do not have adequate resources to grade all the problems on the problem sets then only some problems will be graded.

If you believe a homework problem was misgraded, please return it to the TA, with a note, within one week of when the problem set is returned. If you think that a quiz or the midterm was misgraded, please return it to the instructor, with a note, within one week of when the exam was returned.

Scheduling of exams. The anticipated schedule for the quizzes and midterm is on the living (lecture-by-lecture) syllabus you will find on line.

Prerequisites. ECS 20 is the prerequisite for this class (Math 108 is recommended and is interchangeable as far as we're concerned). This is a serious prerequisite in the sense that you will not do well in this course if you do not have mathematical maturity consistent with having taken, and understood, ECS 20. In particular, you need to be able to understand and create proofs. While we expect very little in the way of *particular* mathematical background, we do

expect that sort of mathematical maturity. If you are a CS major who has trouble with math, consider taking some or all of your math electives before taking this class.

Text. Michael Sipser, *Introduction to the Theory of Computation*, 2nd edition, Course Technology, 2005. We will cover most of the material of Chapters 0–5, 7. You may use the prior edition of the book.

Problem sets. Homeworks are due on Tuesdays at 4:15 pm in the appropriately-labeled box in #2131 Kemper. Late homeworks will not be accepted.

Much of what one learns in this course comes from trying to solve the homework problems, so please work hard on them. I intend for you to find some of the problems challenging. If you're keeping up with the course and are reasonably creative, you should be able to solve most or all of the problems within a few hours. But a few of the problems you might not be able to get. Don't let this discourage you.

Doing a conscientious job on the homeworks is the best preparation for the exams, and it is essential for mastery of the material.

In preparing homework solutions, you absolutely may *not* consult old problem-set solutions, either the instructor's or another student's, either of this course or someone else's course.

Oddly, many students are more willing to spend long hours hacking in front of a machine than to spend them peacefully thinking beneath an old oak tree. I don't know why. It doesn't make sense.

Your writeups should be clear, terse, and neat. Aim for elegance. Obsess. I encourage you to typeset your solutions. I encourage top students—including anyone bound for graduate school—to use LaTeX, a typesetting program especially good for mathematical material. The elegance you should strive for includes (but goes far beyond) pretty typesetting.

For me, clarity of a solution is as necessary as correctness. Don't be surprised to lose points if you provide a correct solution with a poor writeup; I grade that way, and always encourage my TAs to do so. As with an English paper, please do not turn in a first draft: you need to refine your writeup a time or two, making it shorter, simpler, and cleaner.

If you can't solve a problem, briefly indicate what you've tried and where the difficulty lies. Never try to "fake" a solution. Know what you know and be clear about it. In grading exams you will find that I am not big on partial credit (my ideal would be to give none); get all that you can fully right.

Collaboration. In this class, I *discourage* collaboration on homeworks. Learning the material in this subject is a struggle for which others probably cannot help. When I have asked theoretical computer scientists if they collaborated on theory or algorithms homeworks when they were undergraduates, they answer is invariably no. (Of course you can explain that in a variety of ways.)

All the above said, I do not *prohibit* collaboration—and some students sincerely believe that they learn better with it. If you do collaborate, the manner in which you collaborate will have a profound impact on how much you get out of the homeworks. (which will, in turn, have a

profound impact on how you do on the exams). First, think about the problems and try to solve them on your own. If, after giving a problem some real though, you just can't get it, then you might wish to discuss it with other students, with me, or with the TA.

Academic misconduct. If you discuss problems with anyone, acknowledge him/her/them (*I discussed this problem set with so-and-so*). Also acknowledge any books which you consulted other than your own. Write up problems entirely on your own (even if you discuss a problem with someone else).

Some homework questions will have been used in prior years, either by me or by other professors. *You absolutely may **not** consult old problem-set solutions for this class, nor those of any related classes.* Not "official" solutions dug up from the web, and not those from another student in a prior term. Anyone who produces a writeup that appears to have been influenced by an old problem-set solution will be referred to Judicial Affairs.

If you are having personal or academic problems which are motivating you towards academic misconduct, please come and talk to me, instead. I'm not an ogre. (Then again, most ogres say that.)

Some hints. Most students find this class very abstract and challenging. Some students tell us it is the hardest class that they have ever taken. So let us give you a few hints explaining what we are after and how to do well.

First, I really want you to *think.* Don't try to solve the problems by doing some sort of "pattern matching." It just won't work.

This course is about learning a certain sort of problem-solving skill more than it is about learning any specific material. Keeping this in mind may help put things into better perspective.

Even more than with other courses, you must not get behind. The class keeps building on what has already been done. Don't lose the thread. If you get seriously behind you will probably find it impossible to get back on track.

Be selective in note-taking. Actually, I would suggest that you take no notes at all: the book is good and I follow its development pretty closely, so you ought to be able to just sit back, listen, think, and follow. If you feel you must have notes, you might team up with others and take turns.

If you get involved in a study group, don't let the emphasis degenerate into an attempt to get as many homework problems nailed as possible. The homework points don't much matter, and you'll learn more struggling on your own.

Parting thoughts. This is one of my favorite courses in the CS/CSE curricula. We get at the question of what *is* computation. What could be more interesting or more fun?