Course Overview

Tuesday, January 6th, 2015

Course info.

Website: [https://cseweb.ucsd.edu/classes/wi15/cse105-a](https://cseweb.ucsd.edu/classes/wi15/cse105-a)

Lectures: Tuesdays and Thursdays, 12:30–1:50 PM, in CENTR 216
Sections: Mondays, 6:00–6:50 PM, in CENTR 115
& Wednesdays, 3:00–3:50 PM, in CENTR 106
Final exam: Tuesday, March 17th, 11:30 AM – 2:29 PM, room TBA.

All course material will be posted to the course Website — check it regularly! We will also use Automata Tutor for automata practice, Piazza for discussion, Gradescope for exam grading, and GradeSource for posting your scores.

Staff.

Instructor: Hovav Shacham
E-mail: hovav@cs.ucsd.edu

Teaching assistants: Nathan Speidel Yan Yan
E-mail: nspeidel@eng.ucsd.edu yayan@cs.ucsd.edu

Tutors: Victor Alor Maya Nyayapati

See the course Website for instructor offices and office hours.

Prerequisites. The formal prerequisites for CSE 105 are CSE 12 and CSE 21. One of Math 15B, Math 100A, or Math 103A may substitute for CSE 21.

We expect you to be comfortable with mathematical concepts such as sets, tuples, relations, and functions. You should in particular be familiar with the notion that a set is (or isn’t) closed under some operation. We expect you to be comfortable with manipulating formal mathematical definitions reading and writing formal proofs.

We further expect you to be comfortable with basic data structures, such as trees and graphs; with basic algorithms on those data structures, such as depth-first search and graph reachability; and with basic tools of algorithm analysis, such as big-O notation.

Objectives. Our objectives in CSE 105 are: (1) to define computation formally; (2) to show that there are limits to what is computable; (3) to reason precisely about computation and its limits. Along the way, we will pick up some useful tools (finite automata, regular expressions, context-free grammars) and lay the foundation for the study (in CSE 200) of complexity theory.

Learning outcomes. A successful student will learn the following in CSE 105:

- Recall, understand and manipulate formal tuple definitions for DFAs, NFAs, CFGs, and TMs, and understand implications of these definitions and variants thereof.
- Understand and manipulate language definitions written in “set-builder” notation.
• Understand and generate closure-property proofs for claims of the form “if \( L \) is a regular language, then so is \( f(L) \)” for \( L \) transforming a language into another, and also for other classes of languages than regular languages.

• Understand the meaning and implications of closure properties, especially for the standard language operations; e.g., if two languages are not regular, can/must their concatenation be regular?

• Understand and use reductions for decidability and undecidability, and mapping reductions for recognizability.

• Given a language, determine whether it is decidable or undecidable and prove that claim formally and convincingly.

**Computer use.** Using computers in class can be distracting to you and to other students around you. Accordingly, computers (including laptops, tablets, and phones) may not be used in lecture except with staff permission.

**Textbooks.** There is one required text: Michael Sipser’s *Introduction to the Theory of Computation*, second edition. There is a third edition of Sipser’s book, but we will be using the second edition, which has everything we need and is more affordable. New “international edition” copies of the second edition can be obtained for under $20; see the course Website for details.

The course Website includes a calendar that lists assigned sections from Sipser for each lecture. You should carefully read and make sure you understand these assigned sections. You may test your understanding by solving the “exercises” and “problems” at the end of each Sipser chapter. You should be able to solve every “exercise” without referring back to the material in the chapter. You should be able to solve most “problems” with some thought. The midterms and final will more resemble “problems” than “exercises”; the homework will be a mix of both.

There are also two optional texts: Tom Stuart’s *Understanding Computation* (for a different perspective on the class material) and Miran Lipovača’s *Learn You a Haskell for Great Good!* (for a tutorial on Haskell).

**Homework.** There will be weekly homework. The homework and due dates will be posted to the course Website. Homework will mostly take the form of written problem sets, but some homework will also include designing DFAs using the Automata Tutor site, and some will include programming problems in the Haskell language.

Depending on teaching staff, we may grade the entirety of a homework submission, grade only a subset of the problems, or simply check the submission for completeness. Regardless, you are responsible for all the material covered on the homework.

You may discuss the homework with other students, but you must write up your solutions by yourself. See the “Collaboration policy” section, below, for more details.

**Exams.** There will be two closed-book, in-class midterms, each 50 minutes long. The dates for these midterms are posted on the course Website. There will also be a closed-book, comprehensive final exam in the course’s registrar-assigned final exam slot.

For the final only — not for the midterms — you may bring a single letter-size (8.5” × 11”), *handwritten* summary sheet. If you choose to bring a summary sheet with you to the final exam, you will turn it in together with the final.

There will be no makeups for the midterm exams. If you miss a midterm because of a documented medical emergency (and only for this reason), we will substitute the lower of your other midterm score and your final exam score for the missed midterm. If you miss the final exam because of a documented medical emergency, contact the instructor immediately to arrange a makeup exam. If you miss the final exam for any other reason, you will receive a failing grade in the class.

\[1\text{By you, of course.}\]
Grading. The homework will count for 20% of your class score, each midterm for 20%, and the final exam for 40%. The precise weighing of individual homework can be viewed on GradeSource. There is no fixed mapping from class scores computed according to this formula to letter grades. Participation—in lecture, in section, at office hours, and on Piazza—may affect your final grade at the margin.

You may request a regrade of a homework or midterm no later than two weeks after that homework or midterm is handed back. You should do so in writing, explaining exactly what errors in grading you identified.

Homework preparation and submission. All written homework must be typeset (for example, using \LaTeX) and submitted as a PDF. All homework must be submitted using the class online submission tool. Homework submitted by other means—on paper, by e-mail, etc.—will not be accepted.

Late policy. With homework due weekly, getting behind on the homework deadlines is strongly discouraged. Accordingly, homework late by less than 24 hours will be accepted but assessed a 50% penalty. No homework more than 24 hours will be accepted for any reason.

Policy on academic accommodations. Per UCSD policy on academic accommodations (see \url{https://senate.ucsd.edu/Operating-Procedures/Senate-Manual/Appendices/3}), any requests for accommodations must be presented to the instructor and to the CSE department student-affairs staff within the first two weeks of instruction (that is, by Tuesday, January 20th). Accommodations cannot be made retroactively. All requests must be accompanied by a letter of certification and accommodation recommendation from the Office for Students with Disabilities.

Collaboration policy. You may discuss the homework with anyone in the class. You must then write up the solutions entirely on your own. You must list the name of everyone in the class with whom you discussed the homework in your homework writeup. You must not discuss the homework with anyone outside of class.

If you received permission from the course staff to refer to any books or online resources beyond the ones assigned listed under “Book and Internet use policy,” below, you must cite these sources in your homework writeup.

No collaboration whatsoever is allowed on the midterms or the final exam.

If you are unsure about what constitutes allowed collaboration, ask the course staff!

Book and Internet use policy. You may consult any edition of Sipser’s *Introduction to the Theory of Computation*—the second edition is our required text. You may also, if you wish, consult any edition of Hopcroft, Motwani, and Ullman’s *Introduction to Automata Theory, Languages, and Computation*, though note that their notation may differ from Sipser’s. You may, finally, consult Stuart’s *Understanding Computation*, our optional textbook.

You must not consult the solutions manual for Sipser or for Hopcroft-Motwani-Ullman. You must not consult any other book on automata or computability theory without first getting approval from the course staff.

You can and should use the CSE 105 Piazza forum to ask for help on the homework. You must not consult any Internet resource on automata or computability theory without first getting approval from the course staff.

You must not seek out solutions to specific assigned questions on the Internet, from automata courses offered at other universities, or from automata courses offered in previous quarters at UCSD. You must not ask for help on the homework (or exams!) on Internet forums such as Stack Exchange.

You may consult books and Internet resources for general help with Haskell programming and with \LaTeX without getting approval from course staff. You must note any such resources you consulted on your

---

2For the programming assignments, this means that you must code up your programs entirely on your own.

3Note that we expect that “discussed with” is a symmetric relation.
homework submissions. You must not ask for help on Haskell programming from people outside the class or on Internet forums such as Stack Exchange.

If you are unsure about what constitutes allowed use of outside resources, ask the course staff!

Posting of solutions. You must not post online or otherwise distribute to others your solutions to homework or the exams in this course. This prohibition applies both during and after the quarter. This prohibition specifically covers posting any programming assignments to GitHub or similar sites except in a private repository.

Students violating this policy will be subjected to the academic integrity disciplinary process.

Academic integrity. Academic integrity at UCSD is governed by the Policy on Integrity of Scholarship, https://senate.ucsd.edu/Operating-Procedures/Senate-Manual/appendices/2. Allegations of academic misconduct are handled by the Academic Integrity Office, https://students.ucsd.edu/academics/academic-integrity/.

Cheating will not be tolerated, and any student who engages in forbidden conduct will be subjected to the disciplinary process. The course penalty assessed for any cheating in CSE 105 is a failing course grade. Cheaters may additionally be subject to administrative sanctions.

How to do well in CSE 105. The following advice, reproduced (with slight adaptation) from Mihir Bellare’s CSE 107 syllabus (https://cseweb.ucsd.edu/~mihir/cse107/outline.pdf) applies just as well to CSE 105.

Some students operate in a mode I call random access. You look at the homework (perhaps just before it is due), see that you don’t know how to do it, then scan through the slides to see if you can spot some example that looks similar, and try to use that. If that fails, you might ask for help, saying you do not know how to do the homework.

This random access mode of operation is not likely to work well. Here’s the alternative, which I call sequential access. There is a homework due. Ignore it. Instead, read the slides for the chapter in question, sequentially, beginning to end, and make sure you understand everything there. If you don’t, ask for help. Once you have understand everything, do the homework. It will feel a lot easier.

What’s the difference? If you look at the homework and try to map back to the material, your mapping will be imperfect at best. The understanding needed may not be the obvious one. And an example cannot be understood in isolation. In the sequential mode, you aim to understand the material as a coherent whole. It pays off. […]

Do make use of instructor and TA office hours to ask questions. But here’s one lesson. The students who do well are ones who ask questions about the slides and lecture material, not about the homework. If you have trouble with homework, trace it back to something you don’t understand in the slides, and ask about the latter.

If you feel that you understand lecture material and the slides but can’t do the homework, you have created a contradiction. If you can’t do the homework, then, by definition, you do not understand lecture and slide material. Adopting the attitude that you do understand but cannot do the homework is unproductive. It makes it harder for you to help yourself, and makes it harder for us to help you. Instead, if you can’t do a homework, draw the conclusion that you actually don’t understand the material, even if you think you did. This is better because now you know what you have to do and where you can get help.