CS311
Computational Structures

Tim Sheard & James Hook
Portland State University

Syllabus and Class Preliminaries

Registration Details

CRN   11026 (Sheard).   11027 (Hook)

CS 311   Computational Structures (4 cr)

Times: Tue & Thur 10:00-11:50 (Sheard).

  Place: FAB 40-06

Tue & Thur 14:00-15:50 (Hook).

  FAB 40-06 (Hook)

Contact Details:

- Tim Sheard:
  - Office: Fourth Ave Building (FAB) 120-04
  - Telephone: (503) 725-2410
  - Email: sheard@cs.pdx.edu

- James Hook

  Telephone 503-725-5166

  email hook@cs.pdx.edu

Teaching assistant:

- Caylee Hogg
- Email caylee.hogg@gmail.com
- Office hours: TBA

• Further arrangements to be made as the class progresses.

Exams

- Midterm:
  - Tuesday November 12, 2013
- Final:
  - Tuesday **December 10**, 2013, 10:15-12:05 (Sheard)
  - Monday **December 9** , 2013, 10:15-12:05 (Hook)
    - The University scheduled final exam period is not the same as normal class hours!

Methods of assessment:

Class Exercises Start in class, turn in by Thursday class time	10%
Homework (8 weekly homeworks)	40%
Midterm (April 30, 2013)	20%
Final exam (June 11, 2013)	30%
TOTAL	100%

Exercises

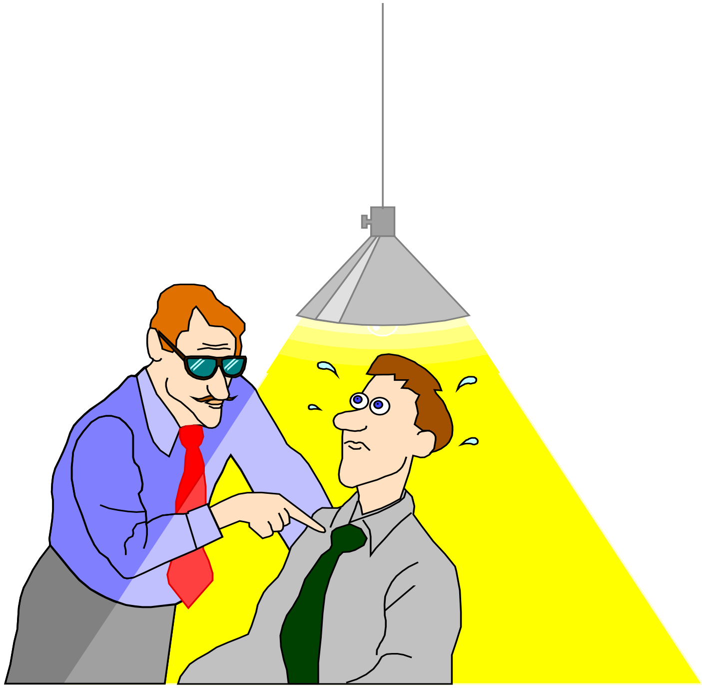
- Exercises are short (less than 1 hour) that are meant to make you think.
- Exercises are assigned on Tuesday and are due via D2L by class time on Thursday.
- Each exercise is worth 1 point.
  - You get full credit for making a good-faith effort to answer the questions.
  - You get 0 points otherwise.

Policies:

- By default, all deadlines are firm.
- We will be as flexible as possible in accommodating special circumstances; but advance notice will make this a lot easier.

Academic Integrity

- We follow the standard PSU guidelines for academic integrity Students are expected to be honest in their academic dealings. Dishonesty is dealt with severely.
- Examinations. Notes and such,
- only as the instructor allows.
- Homework..
  - Discussion is good;
  - Items turned in should be your own
  - individual work. You are encouraged
  - to talk to other people about the
  - homework problems, but you must write
  - up your answers independently. If
  - you're stuck with a problem,
  - please ask for help.



Course Text:

Introduction to the Theory of Computation

(3rd edition)

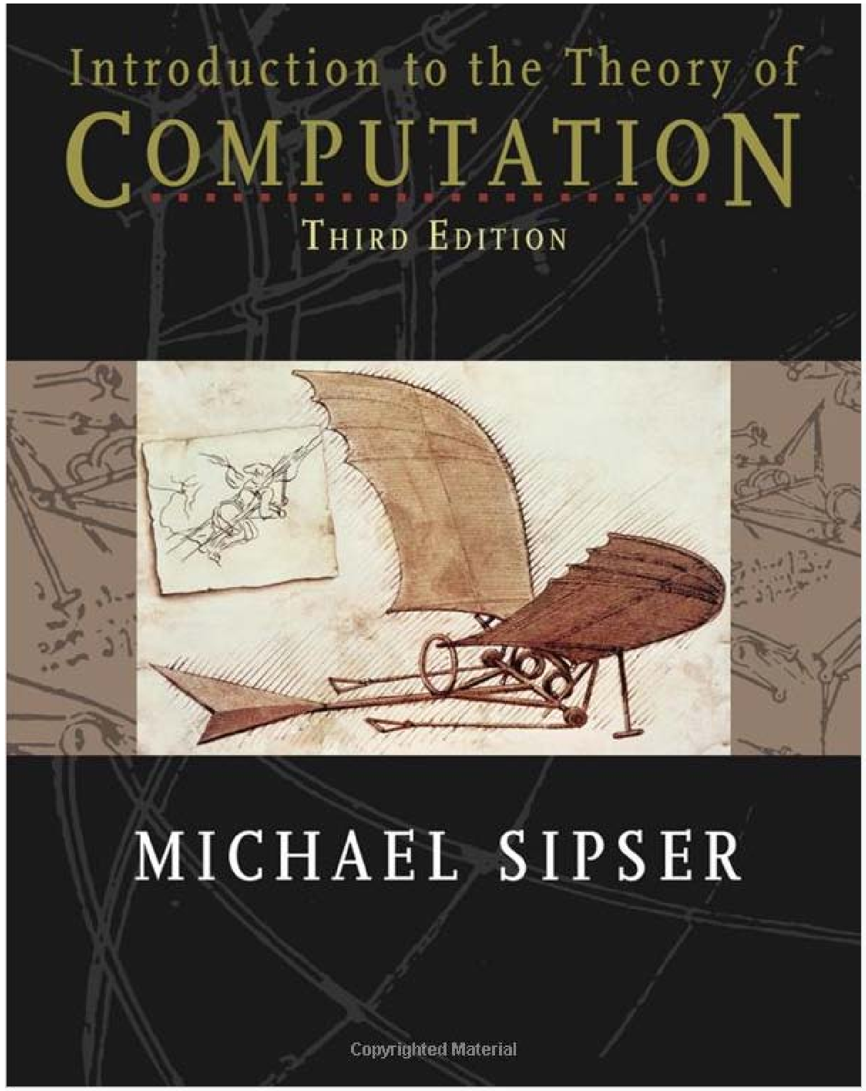
Michael Sipser

ISBN-13   978-1-133-18779-0

Home page of the text book:

http://www-math.mit.edu/~sipser/book.html

It looks like this!



Topics covered

- *Mathematical Preliminaries*
- *Finite State Automata*
- *Non-deterministic Finite State Automata*
- Regular Expressions
- Equivalence of RE DFA NFA
- Regular Language Pumping Lemma
- Context Free Grammars (CFG)
- Push Down Automata (PDA)
- CF pumping Lemma
- Turing Machines
- Church-Turing Thesis
- Decideability
- Diagonalization
- The Halting Problem
- Reducibility
- Complexity

Turning in homework

- You turn in homework via D2L. You may do it by hand, and then scan your assignment into a pdf, or you may use some tool to typeset your homework. Please submit a pdf file.
- Please start every homework with a line with three things: homework number, your name, and your email. It should look like this

- CS311 Homework #1 Tom Smith smith@cs.pdx.edu

The Big Picture

- Computer Science is about computation
- A computational system describes a certain kind of computation in a precise and formal way (DFA, Mealy machines).
  - What can it compute?
  - How much does it cost?
  - How is it related to other systems?
  - Can more than one system describe exactly the same computations?

History

- Early computational systems were based on languages.
- This led to a view of computation that was language related.
  - E.g. which strings are in the language.
  - Is one language a subset (or superset) of another.
  - Can we decide?
  - If we can decide, what is the worst case cost?
  - Are there languages for which the membership predicate cannot be computed?

A Tour of this class

- Languages as computation
  - A hierarchy of languages
    - Regular languages
    - Context Free languages
    - Turing machines
  - A Plethora of systems
    - Regular expressions, DFAs, NFAs, context free grammars, push down automata, Mealy machines, Turing machines, Post systems, and more.
- Computability
  - What can be computed
  - Self applicability (Lisp self interpretor)
  - The Halting Problem

Take aways

- A computational system is like a programming language.
  - A program describes a computation.
  - Different languages have different properties.
  - A language can be analyzed
    - A formal computational system is just data (DFA is a 5-tuple)
    - The structure can be used to prove things about the system
      - What properties hold of all programs?
      - What can never happen?
  - A program can be analyzed
    - A program is just data
    - What does this program do?
    - Does it do the same as another?
    - What is its cost?
    - Is it hard understand?

Why is this important?

- Languages are every where
- Many technologies are based upon languages
  - Parsing, grep, transition systems.
- The historical record has a beauty that is worth studying in its own right.
- Reasoning about computation is the basis for modern computing.
  - What do programs do? What can we say about what they don't do? What do they cost? What systems makes writing certain class of programs easier?
- Computational Systems and Programs are just data.
- Knowing what is possible, and what isn't.