1 Course Link

You can access this course at this link if it is available and you are enrolled in it.

2 Catalog Description

What this course is about: Theory of computation.

Finite automata, regular expressions, grammars, languages, Turing machines, computability, complexity, P, NP and NP-Complete problems.

3 Objectives

- Examine the theoretical basis of computer science. Topics include finite automata, regular expressions, push-down automata, grammars, formal languages, Turing machines, computability, and complexity.
- Learn the real limitations and opportunities in computing. This includes
  - What can and can not be computed (computability),
  - The power of different types of computational systems in terms of what they can compute (automata classification, recursive function theory, Chomsky Hierarchy),
  - What is practically computable and the complexity of solving certain classes of problems (complexity and tractability).
- Appreciate and gain proficiency with rigorous methods, representations, and proof techniques.
- Learn how to apply theoretical concepts to practical problems.
- Prepare for undertaking graduate studies in computer science.

4 Prerequisites

- CS 306 Algorithms and Complexity
5 Course Materials

Required Text:

*Introduction to the Theory of Computation*

Michael Sipser


6 Requirements

You are required to...

- attend class, as attendance and participation factor into your grade.
- read assigned portions of the course materials **before** the class meeting when they will be discussed.
- do homework assignments to enhance your understanding of covered topics.
- explore the terrain of theoretical computer science.
- take online quizzes and tests.
- take a final exam.
- research, write a paper on and present an advanced computability-related topic.

7 Assignments

Homework assignments are of two types: Exercises and Explorations

7.1 Exercises

“Practice through solving problems is essential to learning any mathematical subject.”

(The author, to the educator, in the preface to his book.)

Homework will be given for each of the first few chapters of the book (to be completed at a rate of about one set per section, or per class meeting). This will mostly consist of some Exercises and will be **self-graded**.

“The Exercises review definitions and concepts. The Problems require some ingenuity. Problems marked with a star are more difficult. I have tried to make both the Exercises and Problems interesting challenges.”

(Again quoting Sipser.)
7.2 Explorations

Explorations are programming (and non-programming) assignments that allow you to more fully explore selected course topics. There will be four explorations, about one every two or three weeks.

7.3 Late Policy

Work is due on the day indicated in the schedule. Late work is accepted only if the reason is extraordinary, and acceptance is reached through private and prolonged negotiation. And you must come talk to me in person — no emails, phone calls, nor texts.

8 Assessments

Two tests and a comprehensive final (weighted towards topics in the last part of the course) will be given as outlined in the schedule. The two tests will be administered in the Linux Lab.

The two-part final will be given in class, on the last day of class, and on the university-scheduled testing day, as given on the schedule.

9 Grading

- Assignments: 60%
  - Exercises: 15%
  - Explorations: 20%
  - Research Proposal/Presentation/Paper: 25%
- Assessments: 40%
  - Tests: 20%
  - Finals: 20%

Your weighted percentage will determine your final grade as output from the following function:

```lisp
(defun convert-to-letter-grade (weighted-percentage)
  (let* ((number (ceiling weighted-percentage))
         (tensDigit (/ number 10))
         (onesDigit (mod number 10))
         (index (min (max (- tensDigit 5) 0) 4))
         (letter (substring "FDCBA" index (+ index 1)))
         (sign (if (<= onesDigit 2) "-" (if (>= onesDigit 7) "+")))
           (concat letter (if (and (< number 95) (>= number 60)) sign)))
    )
)
```