Computer Science 351

Welcome To Computer Science 351!

Instructor: Wayne Eberly

Department of Computer Science
University of Calgary

Lecture #1 — Part One

What is CPSC 351?

What is Computer Science 351?

- A course in ***mathematical foundations of computer science*** and ***computability theory*** required for the BSc in Computer Science program at the University of Calgary
- Intended primarily for computer science majors, and other students interested in careers in software development

Expected Background

- ***Mathematical Foundations:***
  - Computer Science 251 — or Statistics 213 and either Mathematics 271 or 273
  - One of Mathematics 249, 265, or 275
  - One of Philosophy 279 or 377
- ***Computer Programming:*** One of Computer Science 219, 233, or 235

Topic #1: Finite Automata and Regular Languages

- We will start by studying and extremely simple (but still useful) kind of abstract machine, and the class of languages that these machines recognize.
- This gives a way to introduce concepts, and techniques, related to computability — without using a machine model that is so complicated that one can get lost in the details.

Topic #2: Turing Machines and Computability

- We will continue with a more complicated kind of abstract machine which is — in a sense — as powerful as any real computer.
- Concepts and techniques from the first part of the course will be extended, and applied, to establish results about what real computers can — and probably *cannot* — do.

Topic #3: Discrete Probability Theory

- CPSC 251 introduced *discrete probability theory*. A study of this will be continued at the end of CPSC 351.
- There is a huge amount of material that *could* be introduced here. Material that is useful for the *design and analysis of algorithms* will be emphasized.

Organization of Course

This is a ***flipped course*** where the following sequence will be used as a topic or problem is considered.

1. *Preparatory Readings and Activities:* To be completed ***before*** a lecture presentation.
2. *Lecture Presentation:* Following a *very* brief review of the preparatory material — with an opportunity to ask questions — the instructor will use this material to solve a problem (or, sometimes, prove a result).

Organization of Course

3. *Tutorial Exercise:* One more problems will be supplied, in a "tutorial exercise", in advance.
   - ***Students will be expected to try to solve these problems — or, at least, think about how to solve them — before the tutorial.***
   - Teaching assistants will act as ***facilitators*** who help students to figure out how problems can be solved and how to evaluate, and improve, the solutions they obtain.
   - These will *not* be "mini-lectures" and solutions will generally ***not*** be presented, in tutorials, at all.
4. *Assessment:* Students will then solve similar problems — by themselves or in small groups — on assignments and tests.

About This Approach

Why Do It This Way?

- "You learn by doing."
- This involves "soft" skills that you will need later on.

Advice:

- Don't delay things any more than necessary.
- ***Participate*** as much as you can.

Things To Do

- Check out the web site:
  - Introduction and Discrete Mathematics Review
  - Introduction to Deterministic Finite Automata
  - More about Course Administration
- Get set up for course communication.
- Start to learn about ***deterministic finite automata***.