Foundations of Computing I
Course Description
Draft of May 18, 2009; (only change since May 9 is formatting)

Structural place in the curriculum:

- 4 credits (3 weekly lectures, 1 weekly section)
- Pre-requisites: 143
- Courses with Foundations of Computing I as a pre-requisite:
  - Programming Abstractions
  - Foundations of Computing II
  - Hardware Design & Implementation
- Taken by: All majors (CS and CE)
- Catalog description: To be determined

Course Overview / Goals:
This course covers topics in logic, discrete mathematics, and aspects of computation that are essential in computer science and engineering. It is a pre-requisite for many courses, and we expect essentially all CSE majors will take the course in the first (or possibly second) quarter in the major. While individual topics are mathematical in nature, the course identifies applications of each topic in computing.

Evaluation and Homework:
The course will have written weekly homeworks, mostly in a "paper and pencil" style, a midterm, and a final.

Possible textbook:
An ideal single textbook has not yet been identified. Many combinations of textbooks would be possible.

Rough topic list:
(Numbers of lectures are approximate.)

- Logic (5-6 lectures)
  - Boolean algebra
  - Truth tables
  - Logical connectives (including multiple notations)
  - Combinational Circuits
  - Two-level logic (CNF/DNF/SOP/POS)
  - Predicates
  - Quantifiers; Translation to and from English
  - Inference: Boolean Algebra Laws and duality, Rules for Implication
  - Proof techniques: by enumeration, direct proof, by contrapositive, by cases, by contradiction

  Applications: One time pads, formal specification, etc.
- Sets, Functions, Relations (2-3 lectures)
  - Boolean Algebra of sets
  - Cartesian Product

- Power Set
- Representation as bit strings
- Functions: 1-1,onto, bijection
- Relations:
  - Equivalence relations: reflective, symmetric, transitive
  - Binary Relations == directed graphs.
  - Transitive relation ~ paths (?)
  - Symmetric relations == undirected graphs

  Applications: To be determined.

- Induction (6 lectures)
  - Induction (1 lec)
  - Strong induction (1 lec)
  - Structural induction (1 lec)
  - Use of induction to prove correctness and analyze simple algorithms (e.g., merge sort) (1 lec)

  Applications: (2 lec)
  - Solving simple recurrence formulas via guess & verify
  - Strings and regular expressions
  - Context-free grammars (?)
  - Properties of Binary Trees
  - Correctness of computations
  - Program Invariants

- Basic Arithmetic Properties: (4 + 1(RSA) lectures)
  - Binary and Base conversion
  - Primes, Factorization
  - Modular addition, multiplication
  - GCD/Extended Euclid's algorithm and modular inverse (division)

  Applications: 2's complement representation (5 minute proof that it works properly), Simple hashing, RSA (using and proving Fermat's Little Theorem/Euler's Theorem)

- Algebra, polynomials, error-correcting codes (Optional: 3 lectures)

- Models and Computability (8-9 lectures):
  - Finite State Machines: (4 lectures)
    - Deterministic Finite Automata with output (Moore machines only)
    - Pattern matching using DFAs
    - Simple argument that DFAs cannot represent some functions e.g. $\{0^n 1^n : n >= 0\}$
    - Minimization (Optional)

    Applications: Pattern matching using DFAs, Adventure Game(?), Simple Controller Design Example (?).
  - Undecidability: (4-5 lectures)
    - Infinite cardinalities, dovetailing, diagonalization
    - Undecidability of Halting Problem for Java Programs

* Reductions (informal): Undecidability of "Hello World"
* Church-Turing Thesis and what this means for any language/computation model
* What is a proof? Goedel's incompleteness theorem (Optional)

Correspondence to Old Curriculum:
This course is closest in content to 321. It also contains material currently in 322 and 370.

Topics previously discussed in 321 not in Foundations I:

- Graphs: Basic terminology of nodes, vertices, edges would be in Foundations of Computing I in conjunction with relations or state machines but not the rest. Mostly covered in Programming Abstractions in the context of basic algorithms and data structures. Pure graph theory would no longer be covered in the core.
- Relations: Closures and antisymmetry not covered.
- Counting and Probablity: In Foundations of Computing II

Through its coverage of boolean logic, circuits, and finite state machines, Foundations of Computing I draws a significant amount of material currently covered in 370. Remaining material from 370 would no longer be in the CS required core but would mostly be in the CE required core as part of Hardware Design and Implementation.

The material on finite state machines and undecidability is currently part of 322. Other material in 322 (such as nondeterministic finite automata, equivalence of finite automata with regular expressions, properties of context-free grammars, pushdown automata, and Turing-Machine details) would not be in the required core for either CS or CE. This may necessitate some small changes in 401 and 431.

Courses having Foundations I as a pre-requisite:

- Hardware Design and Implementation: The material on Boolean logic, circuits, and finite-state machines is essential background and represents a significant portion of our current digital design course.
- Programming Abstractions: Foundations of Computing I develops the general ideas for formal reasoning as well as the key notions for reasoning about recursively defined objects that are essential for understanding the properties of data structures.
- Foundations of Computing II: The reasoning and general mathematical maturity developed in Foundations of Computing I will be an essential background for Foundations of Computing II.