CS 2233 Discrete Mathematical Structures – Fall 08

Material relevant for the Final Exam

- Homeworks 1–8
- **1.1–1.4 Logic:**
  - Propositions and operators (and, or, not, implication, etc.), truth tables
  - Predicates and quantifiers (for-all, exists), nested quantifiers
  - Translating a formula (with or without quantifiers) into English, and translating an English sentence into a formula
  - Equivalences of formulas (e.g., De Morgan's Laws)
  - **NOT:** Rules of inference (1.5)
- **1.6–1.7 Proofs**
  - Different types of proofs (see handout; e.g., direct proof, proof by contradiction, proof by cases)
  - Know how to apply different proof techniques to prove a theorem. (This includes, as a first step, translating an English statement into a propositional formula.)
  - Know how to disprove a theorem (e.g., find a counterexample to disprove a for-all statement)
- **2.1–2.4 Sets, Functions, Sequences, Summations**
  - Sets: Definition, operators (intersection, union,...), power set, cartesian product
  - Functions: Definition of a function, one-to-one/injective, onto/surjective, one-to-one correspondence/bijection, inverse, composition, graphs
  - Sequences: Definition (functions with domain $\mathbb{N}$)
  - Summations: Know how to read the $\sum$ symbol, arithmetic series, geometric series, index substitution
- **3.1-3.3 Algorithms and Complexity:**
  - O, Ω, Θ. (E.g., use definitions to show that $4n + 5 \in O(n^2)$.)
  - Code snippets
- **4.1-4.2 Induction**
  - Weak and strong induction
  - **NOT:** Program correctness and loop invariants (4.5)
- **4.3, 7.1 Recursive Definitions and Recurrence Relations**
  - Recursive functions, sequences, and algorithms
    Know how to develop a recursive solution (i.e., function, sequence, or algorithm) for a problem. (E.g., recursive function for 2^n, or a recursive definition for the sequence $1, 5, 9, 13, 17, \ldots$.)
  - Every recursive definition has a base case and a recursive case.

  - Understand all recursive examples (Fibonacci, $n!$, Towers of Hanoi)
  - Other recursive definitions (sets, arithmetic formulae)

- Recursive algorithms, and solving divide-and-conquer runtime recurrences (Handout from CLRS book, and parts of 7.3)
  - Divide and conquer examples: Mergesort, recursive squaring
  - **NOT:** Develop a divide and conquer algorithm
  - Extract runtime recurrence from a recursive algorithm
  - Solve the runtime recurrence:
    - Generate a guess using either the expansion method or the recursion tree method (knowing any one method is enough)
    - Big-Oh induction
    - Master Theorem (The theorem itself will be given on the test, so you don't need to memorize it.)

- **NOT:** Solving linear recurrence relations (7.2)

- **8.1, 8.2, 8.5 Relations**
  - Definition of binary relations and of k-ary relations, binary relation *on* a set
  - Properties of relations (reflexive, symmetric, antisymmetric, transitive)
  - Equivalence relations, mod, equivalence classes
  - **NOT:** Combining relations (union, intersection), databases and relations

- **9.1–9.3, 10.1 Graphs and Trees**
  - Definition of graph (undirected, directed), terminology (vertices, edges, degree, adjacent, incident, ...)
  - Handshaking theorem (for undirected and for directed graphs)
  - Representation of graphs: Adjacency matrix, adjacency lists.
  - **NOT:** Special types of graphs (bipartite, complete,...), new graphs from old, graph isomorphisms, incidence matrix
  - Definition of trees as acyclic connected undirected graphs, cycles, connectivity
  - Rooted trees and notation (e.g., parent, children, height, descendants, internal vertex, leaf), k-ary trees, full trees
  - Proofs (e.g., induction) on trees and graphs

- **12.1 Languages and Grammars**
  - Definitions of languages and grammars (vocabulary, terminal symbols, productions, etc.)
  - Productions and derivability
  - Backus Naur Form
  - **NOT:** Derivation trees, types of grammars
  - **NOT:** Finite State Machines (12.3)

The Final is on Friday December 12 from 7:30am until 10am in the classroom. It is closed-book and closed-notes, but you are allowed to bring one cheat sheet (a whole one-sided letter page). The Final will contain an exact copy of one question from the first midterm and the second midterm each.