First Exam Study Guide
CISC 471 Compiler Design
Spring 2008

1. **References**
    a. Classtime notes and slides from start of course through March 27
    b. Readings listed on schedule for topics covered in class so far
    c. Project 1 deliverables 0-6
    d. Quizzes
2. **Topic Coverage**
    a. overall compiler phases and the context of a compiler and related tools
    b. lexical specification: regular expressions to specify lexemes for a token
    c. implementation of a lexical analyzer using flex
    d. NFA construction from regular expressions
    e. error detection and recovery in lexical analysis and parsing
    f. syntax specification: context free grammars to specify syntactic units of a language
    g. terminology of lexical analysis and parsing – derivation, ast, leftmost and rightmost derivations, …
    h. problems with grammars – what it means to have an ambiguous grammar
    i. grammar rewriting to attempt to remove ambiguity in expression grammars (associativity and precedence)
    j. top-down parsing: getting the grammar in the right form (left factoring, eliminating left recursion)
    k. Building a recursive-descent (top-down) parser
    l. Operation of a table-driven LL(1) top-down parser
    m. Operation of a bottom-up LR parser
    n. Shift-reduce and reduce-reduce conflicts – what do they signify
    o. Semantic analysis – programming problems not found during parsing
    p. Attribute grammars – terminology, understanding how values get evaluated, limitations that make ad hoc techniques appealing
3. **Format of Exam**

The exam is closed book, closed neighbor and you will have the full class period to work. In general, the exam will be a combination of testing your basic knowledge and understanding of the concepts covered in class and application of the concepts. The questions will most likely be of the form:

- Short answer.
- Writing or reading regular expressions.
- Drawing a DFA for a regular expression
- Understanding of a flex-like specification.
- Writing a context-free grammar for desired syntax
- Rewriting context-free grammars to eliminate left recursion or common prefixes
- Identifying problems in context-free grammars.
- Deriving strings and constructing parse trees.

- Top-down parsing methods: (writing part of a recursive-descent parser, showing the operation of a table-driven LL(1) top down parser
- Bottom-up parsing: showing the operation of a table-driven bottom-up parser
- Attribute grammars: decorating a tree with attribute values given an attribute grammar; identification of synthesized and inherited attributes of an attribute grammar;
- showing understanding of terminology related to these concepts

How to Study

Review notes and slides; practice performing some of the activities above, reviewing what you did on the project, using the textbook as backup for understanding.