First Course Handout

Course Title: Compiler Design
Course No: CS 335
Credits: 3-0-3-12

Prerequisite:

- ESC101, ESO207/CS210, CS220, CS340

Lecture Hours: MWF 9-10:00 AM in RM 101
Office Hours: F 10-10:50 PM in KD 302

Course Objective: The objective of the course is to learn useful concepts to understand, design, and modify compilers for programming languages.

This course will involve both pen-paper and programming assignments.

The course project will require you to apply the concepts learned in the class to build a prototype compiler. You will be required to implement various phases of a compiler and perform an experimental evaluation of your implementation. The project will be done in groups.

Course Contents: The following is a tentative list of topics that we will cover during the course.

- **Overview of Compilation**: analysis-synthesis model of compilation, various phases of a compiler, tool-based approach to compiler construction.
- **Lexical Analysis**: interface with input, parser and symbol table, token, lexeme and patterns, difficulties in lexical analysis, error reporting, implementation, regular definition, transition diagrams, Lex/Flex.
- **Syntax Analysis**: CFGs, ambiguity, associativity, precedence, top-down parsing, recursive descent parsing, transformation on the grammars, predictive parsing, bottom-up parsing, operator precedence grammars, LR parsers (SLR, LALR, LR), Yacc/Bison/ANTLR.
- **Syntax-directed Definitions**: inherited and synthesized attributes, dependency graph, evaluation order, bottom-up and top-down evaluation of attributes, L- and S-attributed definitions.
- **Type Checking**: type system, type expressions, structural and name equivalence of types, type conversion, overloaded functions and operators, polymorphic functions.

- **Runtime Systems**: storage organization, activation tree, activation record, parameter passing, symbol table, dynamic storage allocation.
- **Intermediate code generation**: intermediate representations, translation of declarations, assignments, control flow, Boolean expressions and procedure calls, implementation issues.
- **Code generation and instruction selection**: issues, basic blocks, flow graphs, register allocation, code generation, DAG representation of programs, code generation from DAGs, peephole optimization, code generator generators, and specifications of the machine.

Evaluation:

Assignments	12%
Quizzes	8%
Midsem	20%
Term Project	35%
Endsem	25%

- This is a tentative allocation and might change slightly

References:

1. A. Aho, M. Lam, R. Sethi, and J. Ullman. Compilers: Principles, Techniques, and Tools, 2nd edition.
2. K. Cooper and L. Torczon. Engineering a Compiler, 2nd edition.
3. A. Appel. Modern Compiler Implementation in Java, 2nd edition.
4. M. Scott. Programming Language Pragmatics, 4th edition.