CSC 333: Languages and Machines

Catalog Description:
Prerequisite: "C" or better in CSC 232; and CSC 320 or concurrent enrollment; and MTH 215 or MTH 315. A study of two classes of languages: formal languages (regular, context-free, and computable) and their associated machines (finite automata, pushdown automata, and Turing machines); and programming languages, including the essential features of imperative, functional, object-oriented, and logic programming languages, together with their design and implementation on modern computers and virtual machines. Includes a substantial number of programming assignments, most of which are language interpreters or machine simulators, to be implemented using a functional programming language that emphasizes recursion and higher-order functions. As a semester project, students must research a new programming language and use it to write a nontrivial application. 4(4-0) F,S

This is essentially a terminal course. It is a "maturity" prerequisite for CSC 482, the senior seminar course, in which the CS Major Field Test is administered.

Optional Texts:
Programming Language Pragmatics 3e, Michael L. Scott, Morgan Kaufmann, 2009, https://www.cs.rochester.edu/~scott/pragmatics

Major Topics (including information for course sequence or transition)
1. Regular languages
   a. Regular expressions
   b. Deterministic and nondeterministic finite automata
   c. Closure properties
   d. Regex engines: backtracking, NFA, DFA
2. Context-free languages
   a. Context-free grammars, parse trees, and ambiguity
   b. Writing unambiguous CFGs for expressions
   c. EBNF and recursive-descent parsing
   d. Deterministic and nondeterministic pushdown automata
   e. Closure properties
3. Turing machines
4. Decidable, undecidable, and semidecidable problems
5. Overview of compilation from source to native code
6. Data types and their representation
7. Compilation (for MIPS or stack VM)
   a. Literals, variables, and simple operators
   b. Conditionals: and, or, if
   c. Switch statements
   d. Loops
8. Parameter passing modes
9. OOP concepts and implementation
   a. Classes, objects, and virtual methods
   b. Multiple inheritance vs. (single inheritance + interfaces)
   c. Delegation
10. Implementation of functions
    a. Activation records
    b. Dynamic links, static links, and closures

    c. Calling sequences, function prologue and epilogue
11. Implementation of exception handling
12. Garbage collection
    a. Roots and tracing live data
    b. Properties: conservative, moving, generational, etc.
    c. Examples: mark-and-sweep, semispace
13. Optimization
    a. Basic blocks
    b. Local optimizations: strength reduction, CSE, etc.
    c. Loop optimizations: unrolling, hoisting, etc.
    d. Register allocation by graph coloring
    e. Tail-recursion elimination

Student Outcomes Assessed in CSC 333

A. Students will attain an ability to apply knowledge of computing and mathematics appropriate to the discipline
B. Students will attain an ability to analyze a problem, and identify and define the computing requirements appropriate to its solution
C. Students will attain an ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs
I. Students will attain an ability to use current techniques, skills, and tools necessary for computing practice

CAC Characteristics Enabled But Not Assessed in CSC 333

None

Table 1. Student Outcomes assessed by CSC 333

CSC 333 Student Outcomes	CSC 333 Performance Indicators	CSC 333 Assessment Goals
CSC 333 contributes to SO A: Students will attain an ability to apply knowledge of computing and mathematics appropriate to the discipline	**PI 333-1a:** Construct a regular expression **PI 333-1b:** Construct a finite automaton **PI 333-1c:** Construct a context-free grammar **PI 333-1d:** Construct a Turing machine	**PI 333-1a:** >= 80% correct **PI 333-1b:** >= 80% correct **PI 333-1c:** >= 60% correct **PI 333-1d:** >= 60% correct
CSC 333 contributes to SO B: Students will attain an ability to analyze a problem, and identify and define the computing requirements appropriate to its solution	**PI 333-2:** Classify a language as regular, deterministic context-free, context-free, or not context-free	**PI 333-2:** >= 60% correct
CSC 333 contributes to SO C: Students will attain an ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs	**PI 333-3:** Write a program that implements one component of a compiler	**PI 333-3:** >= 70% of students have a passing score
CSC 333 contributes to SO I: Students will attain an ability to use current techniques, skills, and tools necessary for computing practice	**PI 333-4:** Learn a new programming language well enough to implement the solution to an ACM Programming Contest problem	**PI 333-4:** >= 60% of students have a passing average score