CSC 390  Compiler Construction  4 cr.

Catalog description:
The fundamental problems in the design and implementation of programming language processors are studied. Language syntax and semantics, parsing, implementation techniques such as recursive descent and backtracking, code generation, optimization, and error diagnostics are covered. Concepts are illustrated through extensive programming assignments. Three lecture hours per week and three hours of scheduled laboratory per week, plus extensive laboratory work outside of class.

Prerequisites:  CSC 260.

Goals:
This course is intended to give students an in-depth introduction to programming language processors as an important subdiscipline of computer science and as an example of a problem solution that has achieved orders of magnitude improvement through the rigorous and methodical study of the problem.

CG01:  understand the environment and basic features of a compiler;
CG02:  understand the use of grammars in specifying a language;
CG03:  be familiar with the phases of compilation and how they are implemented;
CG04:  understand how object oriented programming techniques are applied to a compiler.

Objectives:
Upon successful completion of the course, a student will have:

CO01:  learned to analyze and describe the compiler environment;
CO02:  learned to specify, analyze and transform grammars describing programming languages;
CO03:  learned the concepts of the various compiler phases, and analyzed and modified implementations of compiler phases;
CO04:  programmed a simple compiler or fragment, and extended a larger complete compiler.

<table>
<thead>
<tr>
<th></th>
<th>PO1</th>
<th>PO2</th>
<th>PO3</th>
<th>PO4</th>
<th>PO5</th>
<th>PO6</th>
<th>PO7</th>
<th>PO8</th>
<th>PO9</th>
<th>PO10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO01</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO02</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO03</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO04</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Topics:
- Introduction:
  - compilers, interpreters, assemblers
  - single-pass vs. multiple-pass compilers
- Overview: lexical analysis, parsing, code generation
- Representing computer languages:
  - EBNF, grammars, parse trees, syntax diagrams (or syntax trees)
  - Example: a simple compiler (e.g., a recursive-descent expression compiler)
- Scanning and tokenizing
- Grammars (especially context-free grammars)
- Parsing (top-down vs. bottom-up)
- LL parsers
- Symbol tables
- Semantic processing
- Storage management
- Handling types and variables
- Processing expressions
- Translating control structures
- Translating records
- Translating arrays
Assignments:
A number of written assignments will involve practice with EBNF, parse trees, syntax diagrams, etc. Programming assignments will involve the design and implementation of portions of a compiler for a subset of a specific programming language such as Java or C. Consult the instructor for the language(s) to be studied and used in a given semester.

Examinations:
Normally two examinations are given in class at the one-third and two-thirds points of the course. There is also a comprehensive written final examination.

Grades:
The course grade will be determined using the following approximate weights: lab assignments - 10%, programs - 30%, exams - 50%, homework - 10%.

<table>
<thead>
<tr>
<th></th>
<th>Lab Assignments</th>
<th>Programming Assignments</th>
<th>Homework</th>
<th>Exams</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO01</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CO02</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CO03</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CO04</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bibliography:
Aho, Sethi & Ullman  **Compilers, Principles, Techniques and Tools.** (Addison-Wesley, 1988)
Fisher & LeBlanc.   **Crafting a Compiler in C.** (Benjamin-Cummings, 1991)
Kaplan, Randy  **Constructing Language Processors for Little Languages.** (Wiley, 1994)
Metsker, Steven J. **Building Parsers with Java.** (Addison-Wesley, 2001)
Watt & Brown.  **Programming Language Processors in Java,** (Prentice Hall, 2000)