CSC/CPE 431 Programming Languages II

1. CSC/CPE 431 Programming Languages II

2. 

<table>
<thead>
<tr>
<th>credit units</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>contact hours</td>
<td>6</td>
</tr>
</tbody>
</table>

3. Course Coordinator: Aaron Keen

4. Textbook:(and/or other required material) Engineering a Compiler by Cooper and Torczon.

5. a. Course Description: Language principles and design issues: bindings, conversion, parameter passing, and dynamic semantics. Language implementation: intermediate code representation, memory management, code optimization, and code generation. Functional programming languages. 3 lectures, 1 laboratory. Crosslisted as CPE/CSC 431.

b. Prerequisite: CSC/CPE 430.

c. Required/Elective/Selective Elective for CPE, CSC, EE, SE

<table>
<thead>
<tr>
<th></th>
<th>CSC</th>
<th>CPE</th>
<th>SE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elective</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selective Elective</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

6. a. Course Goals/Outcomes

The student will be able to:

This course introduces the student to the design and implementation of a compiler with an emphasis on code generation and optimizations. The course provides an opportunity for small teams of students to work toward the completion of a sophisticated software project.

- Specify and implement intermediate representations of a program.
- Understand the specification of optimization algorithms.
- Given a specification, implement an optimization algorithm.
- Understand and explain the mapping of “high-level” language constructs to assembly code.
- Understand and explain the operation of the run-time stack and parameter passing.

b. How Student Outcomes addressed

(“B” = Basic level, “I” = Intermediate level, “A” = Advanced level)

<table>
<thead>
<tr>
<th>3a</th>
<th>3b</th>
<th>3c</th>
<th>3d</th>
<th>3e</th>
<th>3f</th>
<th>3g</th>
<th>3h</th>
<th>3i</th>
<th>3j</th>
<th>3k</th>
</tr>
</thead>
</table>
7. Major Topics Covered: (number of lecture hours each)

- Intermediate code representation (4)
- Memory management (global, stack, and heap) (2)
- Code optimization (12)
- Register Allocation (2)

Code generation (6)