The compiler is the programmer’s primary tool. Understanding the compiler is therefore critical for programmers, even if they never build one. Furthermore, many design techniques that emerged in the context of compilers are useful for a range of other application areas. This course introduces students to the essential elements of building a compiler: parsing, context-sensitive property checking, code linearization, register allocation, etc. To take this course, students are expected to already understand how programming languages behave, to a fairly detailed degree. The material in the course builds on that knowledge via a series of semantics preserving transformations that start with a fairly high-level programming language and culminate in machine code.

**Course Code:** EECS 322

**When:** Winter

**Instructors:**
Simone Campanoni, simonec@eecs.northwestern.edu
Teaching Assistant: Enrico A. Deiana, enricodeiana2020@u.northwestern.edu

**Course Objectives:**
Students that complete this course should:

1. understand how to efficiently implement a programming language,
2. have an accurate performance model for the primitives in a programming language,
3. have a good sense for the kind of assembly code that a compiler produces
4. be able to maintain and extend a compiler for a programming language

**Assignments:**
In this class, you will learn how to design and efficiently implement a compiler able to generate Intel x86_64 machine code from a high level programming language (a modern C-based language). The work is divided in ten assignments, one per week. Subsequent assignments build on top of the previous ones.

**Materials:**
Recommended book: Modern compiler implementation in C (or Java).
Grading Policy:
Your grade depends on points you will earn on assigned homework and on the code walks. This course is programming project heavy.
You can earn up to 8 points for each homework assigned (8 assignments). Also, you can earn up to 4 points for the panelist experiences. Finally, you earn an extra point if you submit the final compiler on time for the final competition and such compiler passes all tests.
The map between points and grades.

<table>
<thead>
<tr>
<th>Grade</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>12</td>
</tr>
<tr>
<td>A-</td>
<td>10 - 11</td>
</tr>
<tr>
<td>B+</td>
<td>8 - 9</td>
</tr>
<tr>
<td>B</td>
<td>7</td>
</tr>
<tr>
<td>C</td>
<td>6</td>
</tr>
<tr>
<td>D</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>0 - 4</td>
</tr>
</tbody>
</table>

Notes:
This course satisfies the Systems breadth and the project requirement.

Pre-reqs:
EECS 213: Introduction to Computer Systems (or equivalent)

Recommended classes:
EECS 214: Data Structures and Data Management
EECS 321: Programming Languages