1 Course Description and Objectives

Compilers translate code written from one language to another. For example, a C compiler translates programs written in C to binary code that a machine can execute. A compiler itself is a large piece of complex software. For example, GCC is a compiler written for C language. GCC itself is also written in C.

In this course, we will learn how to design such complex software, what are the theories to ensure the translation is correct and successful, and what are the set of tools that can help develop compilers. Students will gain hands-on experience to design a compiler.

Why study compilers?

- It is a foundational subject in Computer Science. Compilers knowledge, such as grammars, parsing, optimizations, is theoretically deep and intellectually interesting.

- It is also a practical subject. There are many programming languages and thus compiler building teams around, e.g., Microsoft Visual Studio, Apple clang compiler, Firefox Rust language, and GCC. Compilers are implementations of programming languages, involving large amount of engineering work and system building.

2 Prerequisite

COM S 331, COM S 342, ENGL 250, SP CM 212

3 References and Resources

Text books (they are not required but highly recommended):


Compiler courses taught by other universities:

1. Stanford cs143 Compilers and Coursera Compilers
2. Rice University COMP 412 Topics in Compiler Construction
3. Berkeley CS 164 Programming Languages and compilers
4. UC Davis ECS142 Compilers
5. Princeton CS320 Compiling Techniques
6. Delaware CISC 672/471 Compiler Design

4 Evaluation

• Class participation (5%): Please attend the class on time and participate the class discussions.
• Exams (35%): Mid-term (15%), Final (20%)
• Written assignments (15%): Answer questions to help understand the theoretical aspects of compilers.
• Programming Assignments (45%): You will work in a team (up to 3 students) to write a compiler for the Cool language.

5 Tentative Topics

1. Introduction
2. Lexical analysis
3. Parsing
4. Semantic analysis
5. Runtime organization
6. Optimizations
7. Code generation
8. Register allocation
9. Garbage collection

6 Course Policies

Absence and late homework policy: We do not grade late homework. Please submit it in time. Your attendance of the class will be considered as a part of Class Participation grade. If you are too sick to attend the class or finish the homework, please send in an email to explain the situation.

Iowa State University’s policy on academic dishonesty: Suspected academic misconduct will be reported to the dean of students office http://www.dso.iastate.edu/ja/academic/misconduct.html