CSC-363 Lecture 01A
Course Introduction

Colin McKinney

19 January 2026

Course Overview

This course explores principles and practices used for designing and implementing compilers and interpreters. Students will build a compiler for a programming language designed for the course. The major stages of compilation will be studied in-depth – lexical analysis, syntax analysis, semantic analysis, and code generation. Additional topics such as advanced parsing techniques and specific compiler-construction tools may be covered at the instructor's discretion.

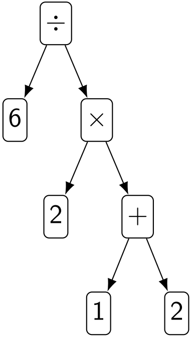
- Syllabus on Canvas: read it!
- Lecture MWF 0800-0850, Studio Thursdays 0800-0915.
- Plan for Studio is to be as interactive as possible, with tangible product due at end of each session
- Problem Sets every week or so; midterm exam during Studio, Final Exam
- Lots of programming. We'll build **two** compilers.

Motivating Example 1: Social Media

$$6 \div 2(1+2) =$$

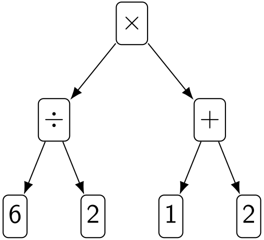
Motivating Example 1: Social Media

Incorrect parse (yields 1)



$6 \div \big(2(1+2)\big) = 6 \div 6 = \mathbf{1}$

Correct parse (yields 9)



$\big(6 \div 2\big)(1+2) = 3 \cdot 3 = \mathbf{9}$

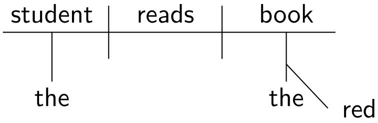
Motivating Example 2: Natural Language

Consider the sentence “The student reads the red book." We can diagram it in a few ways:

Motivating Example 2: Natural Language

Consider the sentence “The student reads the red book." We can diagram it in a few ways:

Reed–Kellogg



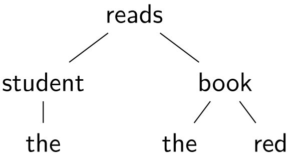
student | reads | book

the

the

red

Syntax Tree



reads

student

book

the

the

red

Motivating Example 2: Natural Language

New problem: let's translate the sentence into French. Can we do a direct word-for-word translation?

- The: *le* or *la*
- student: *étudiant*
- (he/she/it) reads: *lit*
- red: *rouge*
- book: *livre*

- *Le étudiant lit le rouge livre.*
- *Le étudiant lit la rouge livre.*
- *L'étudiant lit la livre rouge.*

From Sentences to Programs

We've seen two forms of translation both today and in CSC-241:

- C to assembly: arithmetic example
- English to French

The *idea* is mostly the same. We first need to decipher the underlying structure in a systematic way, to produce a tree. We then need to use this tree to do the translation.

- Creating the tree is primarily about structure, not meaning. Our sentence might as well have been "The borogove reads the mimsy tove".
- When doing the translation, we needed to understand the structure and meaning, along with the rules of the target language.

History

Everything we've done today feels natural. How else would we do it?

History

Everything we've done today feels natural. How else would we do it?

But someone had to come up with this idea in the first place: especially for computing. The first computers were programmed directly, either by literally rewiring them (ENIAC) or writing in machine language.

History

Everything we've done today feels natural. How else would we do it?

But someone had to come up with this idea in the first place: especially for computing. The first computers were programmed directly, either by literally rewiring them (ENIAC) or writing in machine language.

It's a major conceptual leap to have a computer process structured information written by a human and translate it into machine language.

History

Everything we've done today feels natural. How else would we do it?

But someone had to come up with this idea in the first place: especially for computing. The first computers were programmed directly, either by literally rewiring them (ENIAC) or writing in machine language.

It's a major conceptual leap to have a computer process structured information written by a human and translate it into machine language.

Enter Admiral Grace Hopper.

Admiral Grace Hopper

- Mathematician who taught at Vassar College. Joined the Navy in WW2.
- Worked on Harvard Mark 1 computer during the war.
- After the war, worked on the UNIVAC (first commercial computer).
- "...I decided data processors ought to be able to write their programs in English, and the computers would translate them into machine code."
- This is the same idea we used today, made explicit and systematic.

Looking Forward: Our Central Goal

- Taking human-written programs seriously as *structured objects*
- Making implicit structure explicit
- Eliminating ambiguity
- Systematically translating structure and meaning into machine-executable form

By the end of this course:

- You will understand how computers *read*.
- GCC will no longer be a magic box.