EC3204: Programming Languages and Compilers

Lecture 0 — Course Overview

Sunbeom So
Fall 2024

Basic Information

- **Instructor:** Sunbeom So (소순범)
- **Position:** Assistant Professor in EECS, GIST
- **Expertise:** Program Analysis
- **Office:** EECS C501
- **Office hours:** by appointment
- **Email:** `sunbeomso@gist.ac.kr`
- **Course materials:** https://github.com/gist-pal/ec3204-pl-and-compilers

Warning:
Our goal is not to learn particular programming languages!

Goal

- To study the principles of compiler construction and related programming language theories.
- Deeply understand CS in general by implementing a toy compiler.

Topics (tentative)

- **Lexical analysis:** lexical tokens, regular expressions, finite automata, lexical analyzer generators
- **Syntax analysis:** context-free grammars, top-down parsing, bottom-up parsing, parser generators
- **Semantic analysis:** type checking, verification, static analysis
- **Intermediate representation generation:** syntax-directed translation, three address code, control flow graph, basic blocks
- **Code optimization:** data-flow analysis, semantics-preserving transformation
- **Code generation (optional):** register allocation and assignments, instruction selection, machine code generation

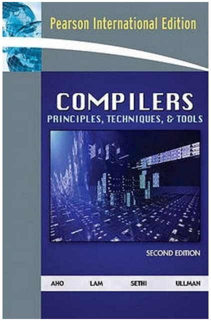
Prerequisites

- **Prerequisites:** To succeed in this course, you are recommended to:
  - have completed of the basic CS courses (e.g., Automata Theory, Data Structures, Algorithms).
  - have extensive experience in computer programming.
- **This is an advanced course.** Complaints like the below ones are difficult to be addressed (sorry!).
  - "Teach me more about the syntax of OCaml language."[1]
  - "Programming assignments are too hard."

[1] I will provide basic guidance, but will not explain every single detail from one to ten.

Lecture Slides & References

- Self-contained slides will be provided.
- "Compilers: Principles, Techniques, and Tools" (Pearson International Edition/Second Edition) by Aho, Lam, Sethi, and Ullman.


Pearson International Edition
COMPILERS
PRINCIPLES, TECHNIQUES, & TOOLS
SECOND EDITION
AHO
LAM
SETHI
ULLMAN

Grading Policy

- Mid-term/Final exam (100 points for each).
- 3-4 programming assignments in OCaml.[2]
- Your final grade will be given as follows.
  1. Assign letter grades based on the exam results (F – A).
  2. If you have completed[3] 2 programming assignments, your grade will be increased by one level (e.g., B+ $\rightarrow$ A, B $\rightarrow$ B+).
  3. If you have completed all the assignments, you are eligible for an A+.
- If all you need is an A or a lower grade under this grading system, it is perfectly fine not to complete any assignments.

[2] `https://ocaml.org/`

[3] passed 90% of testcases provided by the instructor (hidden at the submission time)

Attendance

- **You must attend more than 2/3 of the classes** according to GIST regulations (see No.25 of ER-322-07). Otherwise, you cannot take the exams.
- **As long as you conform to this rule, attendance will not affect your letter grade.** Please avoid sending me emails before your absence.

Academic Integrity

Read Carefully.

- All assignments (i.e., writing code) must be your own work. **No discussions are allowed**.
  - You should not share/show your code.
  - You should not post your code on public websites.
  - You should not modify other students' code.
- **Cheating on assignments will result in an F.**
- I will have a one-on-one meeting with each student under suspicion.
- If you fail to prove your integrity in the meeting,[4] I will consider that you cheated on your assignments, even if you do not admit your cheating.
- Your grade is not negotiable. Do not send me an email for it after the grade notification.

By registering for this course,
I will assume you agree with this policy.

[4] e.g., failing to answer my questions about the details of your submissions