

COSE212: Programming Languages

Lecture 0 — Course Overview

Hakjoo Oh
2015 Fall

Basic Information

Instructor: Hakjoo Oh

- **Position:** Assistant professor in Computer Science and Engineering, Korea University
- **Expertise:** Programming Languages and Compilers
- **Office:** 616c, Science Library
- **Email:** `hakjoo_oh@korea.ac.kr`
- **Office Hours:** 1:00pm–3:00pm Mondays and Wednesdays (by appointment)

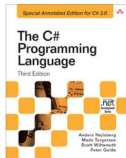
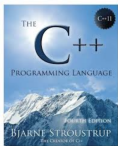
TA:

- Kwonsoo Chae
- Email: `kwonsoo.chae@gmail.com`

Course Website:

- <http://pr1.korea.ac.kr/~hakjoo/courses/cose212/2015/>
- Course materials will be available here.

Programming Languages



Objectives

Learn fundamental principles necessary to deeply understand the programming languages:

- essential concepts of modern programming languages
- how to specify and implement programming languages

Topics

- **Part 1 (Preliminaries):** inductive definition, programming in functional, typed, and higher-order language languages

Topics

- **Part 1 (Preliminaries):** inductive definition, programming in functional, typed, and higher-order language languages
- **Part 2 (Basic concepts):** syntax, semantics, naming, binding, scoping, environment, interpreters, states, side-effects, store, reference, mutable variables, parameter passing

Topics

- **Part 1 (Preliminaries):** inductive definition, programming in functional, typed, and higher-order language languages
- **Part 2 (Basic concepts):** syntax, semantics, naming, binding, scoping, environment, interpreters, states, side-effects, store, reference, mutable variables, parameter passing
- **Part 3 (Advanced concepts):** type system, typing rules, type checking, soundness/completeness, type inference, polymorphism, modules, module procedures, typed modules, objects, classes, methods, inheritance, typed object-oriented languages

Textbook

- Essentials of Programming Languages (Third Edition) by Daniel P. Friedman and Mitchell Wand. MIT Press.



- Self-contained slides will be provided.

Prerequisites

- Programming experiences in at least two languages
- Basic courses in CS: Introductory programming courses, data structures, theory of computation

Grading

- Homework – 40%
 - ▶ 0–1 pencil-and-paper assignments
 - ▶ 4–6 programming assignments
- Midterm project – 25%
 - ▶ replacement of Midterm exam
- Final exam – 30%
- Attendance and participation – 5%

Assignment policy:

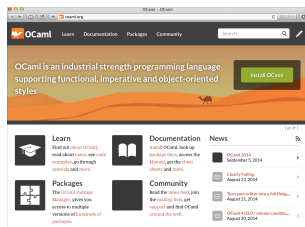
- No late submissions will be accepted.
- All assignments must be your own work.
 - ▶ Copying gets you 0 for the entire HW score.

Programming Assignments in ML

ML is a family of programming languages including SML, OCaml, F#, etc.

- Support higher-order, strict, mostly pure, and typed, with algebraic data types.
- Inspired the design of many modern programming languages.
- Suitable for implementing language processors.
- A good deal of syntax.

We will use OCaml:



Schedule (tentative)

Weeks	Topics
1	Introduction
2	Inductive Definition
3	Functional Programming in ML
4	Scoping, Binding, and Procedures
5	Scoping, Binding, and Procedures
6	Scoping, Binding, and Procedures
7	States
8	Parameter Passing
9	Mid-term project (no class)
10	Type Checking
11	Type Inference
12	Type Inference
13	Modules
14	Modules
15	Objects and Classes
16	Final exam

Homework 0: Hello World

- 1 Install OCaml in your system from <http://ocaml.org>

Homework 0: Hello World

- 1 Install OCaml in your system from <http://ocaml.org>
- 2 Write the following code and store it as `hello.ml`.

```
print_string ("Hello World!\n")
```

Homework 0: Hello World

- 1 Install OCaml in your system from <http://ocaml.org>
- 2 Write the following code and store it as `hello.ml`.
`print_string ("Hello World!\n")`
- 3 We can run the program with three different ways:

- 1 Using the REPL:

```
$ ocaml
OCaml version 4.01.0
```

```
# #use "hello.ml";;
Hello World!
- : unit = ()
```

- 2 Using the interpreter:

```
$ ocaml hello.ml
Hello World!
```

- 3 Using the compiler:

```
$ ocamlc hello.ml
$ ./a.out
Hello World!
```