Types and Programming Languages

Kwanghoon Choi

Software Languages and Systems Laboratory
Chonnam National University

Week 0

This class

The study of type systems and and of programming languages from a type-theoretic perspective

Potential audiences

- mature undergraduates/graduates from all areas of computer science who want an introduction to key concepts in the theory of programming languages

An extensive introductory material and a wealth of examples, exercises, and case studies.

Materials

Textbook

- Types and Programming Languages by Benjamin C. Pierce
- http://www.cis.upenn.edu/∼bcpierce/tapl

Typecheckers present in this course are available in concrete implementations written in O'Caml

Goals: Theory

Core topics

- Basic operational semantics for programming languages
- The untyped lambda calculus
- Simple type systems
- Associated proof techniques to show its type safety
- Simple extensions: the unit type, let bindings, pairs, tuples, records, sums, variants, general recursion, lists
- More extensions: references, exceptions.
- Extended type systems for the extended lambda calculi
- To prove the extended type safety using the same proof techniques

Required Background

We assume no preparation in the theory of programming languages.

But readers should start with a degree of mathematical maturity - in particular, rigorous undergraduate coursework in discrete mathematics, algorithms, and elementary logic

It would be good for readers to be familiar with at least one higher-order functional programming language (Scheme, ML, Haskell, etc.), and with basic concepts of programming languages and compilers (abstract syntax, BNF grammars, evaluation, abstract machine, etc.)

Goal: Practice

As well as the theoretical topics, students will learn a functional programming language called O'Caml.

O'Caml is a functional programming language with object-oriented styles.

- https://ocaml.org

Tutorial: Basic language concepts and how to install

- https://dev.realworldocaml.org/

No required preliminary background is needed.