Lecture 1: Introduction

Polyvios Pratikakis

Computer Science Department, University of Crete

Type Systems and Static Analysis
General Information

Class code: CS490.40
Instructor: Polyvios Pratikakis
Email: polyvios@ics.forth.gr
Office hours: Mondays 12:15–14:00, K-327
Mailing list: subscribe hy490-40-list
Webpage: http://www.csd.uoc.gr/~hy490-40
Content

- An introduction into the research field of programming languages
- Formal systems for describing and understanding programming languages
- Programming language features and semantics
- Static analysis: techniques for automatically reasoning about programs
- Functional programming
Goals

1. Learn functional programming in OCaml
2. Study lambda calculus
3. Use it to describe functional and imperative features of programming languages
4. Study language semantics as a way to describe the meaning of programs
5. Study static analysis techniques,
   - Type systems
   - Data flow analysis
   - Alias analysis
6. Learn program verification
   - Hoare logic
What you need to do

- Two lectures per week
- Five homework assignments during the first half of the semester
  - Small programs in Ocaml
  - Improve understanding of material
  - Personal work (no teams)
  - Expected to take about 4–8 hours per assignment
  - Homeworks will be graded automatically
    - No partial credit for code that does not compile or work
- One mid-term exam
  - Exam material is everything covered in lectures until the mid-term
- One term project
  - Learn and use LLVM (C++)
  - Implement a static code analysis and transformation
  - Grade based on (i) implementation, (ii) project presentation, (iii) report
- Final exam
  - Exam material is everything taught during the term
Grading

- Grade consists of:
  - Homeworks: 3 points
  - Project & presentation: 3 points
  - Mid-term exam: 3 points
  - Final exam: 3 points

- Requirement for passing grade is 50% on the final exam

- There are two bonus points (max grade is 12/10)
  - Scores over 10 will be truncated to 10
Books and other reading material

- Types and Programming Languages, B. Pierce
  - [http://www.cis.upenn.edu/~bcpierce/tapl/](http://www.cis.upenn.edu/~bcpierce/tapl/)

- Logic in Computer Science: Modeling and Reasoning about Systems, Huth and Ryan

- Principles of Program Analysis, Nielson, Nielson, and Hankin

- Ocaml Resources
  - Main page: [http://caml.inria.fr/](http://caml.inria.fr/)

- Other online PL texts
  - [http://www.cs.uu.nl/wiki/Techno/ProgrammingLanguageTheoryTextsOnline](http://www.cs.uu.nl/wiki/Techno/ProgrammingLanguageTheoryTextsOnline)
Class dependencies

- **Required**
  - http://www.csd.uoc.gr/~hy255
  - http://www.csd.uoc.gr/~hy280

- **Recommended**
  - http://www.csd.uoc.gr/~hy180
Next time

- Introduction to OCaml
- A functional language in the family of ML
- Object oriented
- Supports imperative code
  - Not in this class
- Good for scripting, quick development
- Usually, if it compiles, it works
  - The benefit of type systems!