## Summary

The course introduces the foundations on which programs and programming languages are built. It introduces syntax, types and semantics as building blocks that together define the properties of a program part or a language. Students will learn how to apply these concepts in their reasoning.

## Content

- simple types, lambda-calculus
- normalization, references, exceptions
- subtyping
- recursive types
- polymorphism
- advances features of the Scala type system

## Learning Prerequisites

### Recommended courses

Advanced topics in programming, Compiler construction

### Important concepts to start the course

Functional programming
Basic knowledge of formal languages

## Learning Outcomes

By the end of the course, the student must be able to:

- Argue design decisions of programming languages
- Assess / Evaluate soundness of type systems
- Compose higher-order functions
- Verify progress and preservation in type systems
- Work out / Determine operational equivalences
- Carry out projects of 2-3 weeks duration
- Distinguish valid from invalid proofs
- Implement type systems and operational semantics

## Transversal skills
• Assess progress against the plan, and adapt the plan as appropriate.
• Evaluate one's own performance in the team, receive and respond appropriately to feedback.
• Identify the different roles that are involved in well-functioning teams and assume different roles, including leadership roles.
• Manage priorities.

Teaching methods
Ex cathedra, practical exercises

Assessment methods
With continuous control

Resources
Ressources en bibliothèque
• Types and Programming Languages / Pierce

Websites
• http://lampwww.epfl.ch/teaching/index.html.en