CS-320

Computer language processing

Kuncak Viktor

Cursus	Sem.	Type
Communication systems	BA5	Opt.
Computer science	BA5	Opt.

Language of teaching	English
Credits	6
Session	Winter
Semester	Fall
Exam	During the semester
Workload	180h
Weeks	14
Hours	**6 weekly**
Courses	2 weekly
Exercises	2 weekly
TP	2 weekly
Number of positions	

Summary

We teach the fundamental aspects of analyzing and interpreting computer languages, including the techniques to build compilers. The new title is "Computer Language Processing".

Content

1. Overview, source languages and run-time models
2. Review of formal languages
3. Lexical analysis
4. Syntactic analysis (parsing)
5. Name analysis
6. Type checking
7. Code generation
8. Data-flow analysis
9. Run-time organization and memory management

Keywords

programming language;
compiler;
interpreter;
regular expression;
context-free grammar;
type system;
code generation;
static code analysis

Learning Prerequisites

Recommended courses

Discrete structures
Theoretical computer science
Programming in Scala
Computer architecture I

Learning Outcomes

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

- Design a programming language
- Construct a compiler
- Coordinate development with project partner
- Formulate correctness conditions for compiler
- Estimate time to implement a programming language feature
- Produce a working programming language implementation
- Decide which language features make implementation difficult
- Specify programming language and compiler functionality

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.
- Respect the rules of the institution in which you are working.
- Continue to work through difficulties or initial failure to find optimal solutions.
- Demonstrate a capacity for creativity.
- Take feedback (critique) and respond in an appropriate manner.
- Make an oral presentation.
- Write a scientific or technical report.

Teaching methods

- Ex catedra
- Exercises on whiteboard
- Exercises using dedicated software
- Project work, indepdently and under supervision of assistants

Assessment methods

- 50% Project
- 25% Mid-term quiz
- 25% End-of-term quiz in December

Resources

Bibliography

Andrew W. Appel, **Modern compiler implementation in Java** (or **ML**), Addison-Wesley 1997 (full PDF available from EPFL library)
Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman: **Compilers: Principles, Techniques, and Tools** (2nd Edition, 2006)
Niklaus Wirth: **Compiler Construction,** neat textbook from a prominent classical authority. Freely available **http://www.ethoberon.ethz.ch/WirthPubl/CBEAll.pdf**

Ressources en bibliothèque

- Additionally, all material
- Compilers, principle, techniques and tools / Aho
- Compiler Construction / Wirth
- Modern compiler implementation in Java / Appel

Notes/Handbook

http://lara.epfl.ch/w/cc
Faboulous and gently paced videos: https://www.coursera.org/course/compilers

Prerequisite for

Synthesis, analysis and verification
Advanced compiler construction
Recommended for Foundations of software