

CS-214

Software construction

Kuncak Viktor, Odersky Martin, Pit-Claudél Clément

Cursus	Sem.	Type
Communication systems	BA3	Opt.
Computer science minor	H	Opt.
Computer science	BA3	Obl.
Cyber security minor	H	Opt.
Data science minor	H	Opt.
HES - IC	H	Opt.

Language of teaching	English
Credits	8
Session	Winter
Semester	Fall
Exam	Written
Workload	240h
Weeks	14
Hours	8 weekly
Courses	3 weekly
Exercises	2 weekly
Lab	3 weekly
Number of positions	

Summary

Learn how to design and implement reliable, maintainable, and efficient software using a mix of programming skills (declarative style, higher-order functions, inductive types, parallelism) and fundamental software construction concepts (reusability, abstraction, encapsulation, composition, proofs)

Content

Functional programming:

- Functional programming paradigm
- Recursion
- Evaluation strategies, lazy evaluation, substitution model
- Modularity, data abstraction, representation independence
- Subtyping, inheritance, type classes
- Polymorphism, variance
- Structural induction
- Stateless parallelism, map-reduce, associative operations
- Effects: state, exceptions
- Documentation, tests, specification
- Interpreters and program semantics

Software engineering:

- Specifications: Documentation, requirements, properties
- Verification: Debugging, tests, monitoring, property-based testing, proofs
- Modularity: Code evolution and refactoring
- Collaboration: Version control, changelogs

Learning Prerequisites**Required courses**

Any previous programming course

Recommended courses

CS-107 Introduction à la programmation
CS-108 Pratique de la programmation orientée-objet

Important concepts to start the course

Loops, conditionals, variable and type declarations, computing mathematical expressions

Learning Outcomes

- Implement reliable, efficient, modular, and maintainable software
- Identify data types and operations that lead to computational solutions
- Argue that an implemented solution is correct
- Transform programs to change their behavior in a desirable way
- Design and implement data-parallel software using parallel collections
- Make use of type systems and tests to develop reliable software
- Argue that an solution is correct

Teaching methods

- Ex cathedra (live lectures)
- Recorded videos
- Exercise and lab sessions
- Online discussions

Expected student activities

- Attending lectures
- Watching and understanding recorded videos
- Solving exercises individually or in groups
- Completing individual graded programming assignments (labs / mini-projects)
- Completing midterm and end-of-semester exams

Assessment methods

- In-class exams during the semester
- Final exam during the exam session
- Programming assignments (labs)

Supervision

Office hours	Yes
Assistant.e.s	Yes
Forum	Yes

Resources

Virtual desktop infrastructure (VDI)

Yes

Websites

- <https://cs-214.epfl.ch/info/syllabus/>

Moodle Link

- <https://go.epfl.ch/CS-214>

Prerequisite for

CS-320 Computer language processing
CS-311 The Software enterprise - from ideas to products
CS-452 Foundations of software
CS-550 Formal verification