Program	Level	First cycle
	Name of the program	Theoretical Computer Science

COURSE

Course title	**Compilers**			
Course code	Semester	Course status	ECTS	Contact hours (L+AE+LE)
CS345	V	Elective course	4	2+0+2
Lecturer				
Course Goals	The aim of the course is to introduce students with many common procedural languages, as well as representational functional, logic-oriented and object-oriented languages.			
Learning Outcomes	Upon successful completion of this course, students are expected to: - get acquainted with many common procedural languages, as well as representational functional, logic-oriented and object-oriented languages, - understand the components and features of programming languages needed for program development and maintenance of various applications, - study the phases and components of typical programming language translators, - study relevant language theory and understand its use in translation, - to design, develop and test large software projects, using multiple software tools, resulting in a programming language interpreter.			

COURSE CONTENT

- Low-level programming, evolution of major programming languages
- Description of syntax and semantics
- Lexical analysis, syntax analysis
- Names and associations, scope, lifetime, environments
- Primitive data types, fields, other types and type checking
- Arithmetic expressions, Boolean expressions, assignments, mixing and equivalence of types
- Imperative programming and structured programming, Control commands: selection, iteration, branching
- Subroutines and parameter passing, problems that happen with subroutines, implementation of subroutines, implementation of nested subroutines, blocks
- Abstract data types and encapsulation
- Object-oriented programming, examples of object-oriented languages
- Exceptions, event-driven programming, and concurrency

LITERATURE

[1] Keith Cooper and Linda Torczon, Engineering a Compiler, Morgan Kaufman, 2011

[2] Robert Sebesta, Concepts of Programming Languages, Pearson; 10 edition (January 16, 2012)

[3] Alfred Aho, Monica Lam, Ravi Sethi, and Jeffrey Ullman, Compilers: Principles, Techniques, and Tools (Second Edition) Addison-Wesley

[4] Andrew Appel and Jens Palsberg, Modern Compiler Implementation in C (Second Edition), Cambridge University Press

STUDENT WORKLOAD (hours in a semester)

Lectures	30	Tutorial	30	Individual work	40	Total	100

GRADING

Criterion	Maximum points	Minimum points
Midterm exams	40	20
Seminar	20	
Final exam	40	10
Total	100	55

REMARKS

2x20 points written tests, remaining 10 points are earned for work during the semester. 5 homeworks worth 2 points each.