Fakultät für Ingenieurwesen
Facoltà di Ingegneria
Faculty of Engineering

SYLLABUS
COURSE DESCRIPTION YEAR 2024/2025

COURSE TITLE	**Formal Languages and Compilers**
COURSE CODE	76214
SCIENTIFIC SECTOR	INF/01
DEGREE	Bachelor in Computer Science
SEMESTER	2nd
YEAR	2nd
CREDITS	6

TOTAL LECTURING HOURS	60
TOTAL LAB HOURS	20
ATTENDANCE	Attendance is not compulsory but recommended. Non-attending students must contact the lecturer at the start of the course to agree on the modalities of the independent study. Exam modalities for non-attending students are the same as for attending students.
PREREQUISITES	Knowledge of the C programming language. Experiences working in Linux environment is preferrable.
COURSE PAGE	TEAMS of the course and https://ole.unibz.it/

SPECIFIC EDUCATIONAL OBJECTIVES	• Type of course: "caratterizzante" • Scientific area: 'Discipline informatiche" The main objective is to introduce the fundamental notions about formal languages (Chomsky classification of Languages, Regular Languages, Automata, Context Free Grammars) and understand the mechanisms governing the analysis and synthesis of programming languages. Students will learn the most important techniques for the representation and generation of Languages (in particular, regular and context-free languages). Those techniques will be applied to the construction of a compiler for a programming language. During this course the student will learn how to build the different parts of a Compiler with a particular emphasis on Lexical Analyzers (with the use of Lex during the Lab), Parsers (with the use of YACC during the Lab) and basics of code generation.

LECTURER	Alessandro Artale

SCIENTIFIC SECTOR OF THE LECTURER	INF/01
TEACHING LANGUAGE	**Italian**
OFFICE HOURS	During the lecture time span, Office 2.03. To fix an appointment send an email to artale@inf.unibz.it
TEACHING ASSISTANT	-
OFFICE HOURS	-
LIST OF TOPICS COVERED	• Formal language theory • Regular languages: automata, regular expressions, regular grammars • Context free languages (stack machines) • Lexical and syntactic analysis: Lexer specification, top-down and bottom-up parsing • Semantic analysis rules for: type checking, symbol table and control flow • Intermediate code generation
TEACHING FORMAT	Frontal lectures, labs with (programming) exercises, team projects.

LEARNING OUTCOMES	**Knowledge and understanding** • Know the concepts of formal languages, and the techniques of compilation of various high level programming languages. **Applying knowledge and understanding** • being able to develop and construct translators and compilers. **Ability to make judgments** • be able to collect and interpret useful data and to judge information systems and their applicability; • be able to work autonomously according to the own level of knowledge and understanding. **Communication skills** • be able to use one of the three languages English, Italian and German, and be able to use technical terms and communication appropriately. • be able to work in teams for the realization of IT systems. **Ability to learn** • have developed learning capabilities to pursue further studies with a high degree of autonomy.

ASSESSMENT	Project conducted in team and a written exam. In the project part of the exam we will assess the learning outcomes related to the application of the acquired knowledge, the ability to make

	judgments and the communication skills. In fact, the goal of the project is to design a compiler for a small programming language. The project part must be positively evaluated before the written exam. In the written exam there will be verification questions, transfer of knowledge questions and exercises. The learning outcome related to knowledge and understanding, applying knowledge and understanding, and those related to the student ability to learn and the acquired learning skills will be assessed by the written exam.
ASSESSMENT LANGUAGE	Italian
EVALUATION CRITERIA AND CRITERIA FOR AWARDING MARKS	• Project: Compiler Development (30%) • Final Written Exam (70%) Written exam questions will be evaluated in term of correctness and clarity. Project is evaluated in term of quality of the solution: complexity and novelty of the programming language to be compiled, data structures used in implementing the symbol table, depth of the semantic analysis carried on. Note: The project will count for 3 regular consecutive exam sessions.

REQUIRED READINGS	Compilers: Principles, Techniques, and Tools (2nd edition). Alfred V. Aho, Monica S. Lam, Ravi Sethi and Jeff Ullman. Publisher: Addison Wesley, 2007. Introduction to Automata Theory, Languages, and Computation (3rd edition). J.E. Hopcroft, R. Motwani, J.D. Ullman. Addison Wesley, 2007.
SUPPLEMENTARY READINGS	Compiler Construction: Principles and Practice, Kenneth C. Louden. Publisher: Brooks Cole, 1997. Advanced Compiler Design and Implementation, Steven Muchnick. Publisher: Morgan Kaufmann, 1997. Programming Language Processors in Java: Compilers and Interpreters, David Watt and Deryck Brown. Publisher: Prentice Hall, 2000.
SOFTWARE USED	C, YACC, LEX, Linux OS