



## COURSE PRESENTATION FORM – ACADEMIC YEAR 2010/2011

<b>COURSE NAME</b>	<b>Compiler</b>
<b>COURSE CODE</b>	70091 (BSc – DM 509)
<b>LECTURER</b>	Alessandro Artale
<b>TEACHING ASSISTANT</b>	
<b>TEACHING LANGUAGE</b>	English
<b>CREDIT POINTS</b>	4
<b>LECTURE HOURS</b>	24
<b>EXERCISE HOURS</b>	12
<b>TIME SPAN</b>	21.02.2011 - 11.06.2011
<b>TIME TABLE</b>	See <a href="#">Timetable Page</a>
<b>OFFICE HOURS LECTURER</b>	During the lecture time span: Wednesday from 16:00 to 18:00; POS building, Piazza Domenicani 4, office 2.03
<b>OFFICE HOURS TEACHING ASSISTANT</b>	
<b>PREREQUISITES</b>	Formal Languages
<b>OBJECTIVES</b>	In this module students will develop a deeper understanding of Compilers technology. Students will learn the most important techniques for the representation and generation of Languages. Those techniques will be then applied to the construction of a compiler for a programming language. In particular, during this module the student will learn how to build the different parts of a Compiler: Lexical Analyzer, Parser and Code Generation.
<b>SYLLABUS</b>	Introduction to the Notion of a Compiler. Grammars and Lexical Analyzers. Syntax Analysis and Parser construction Top-Down Parser Bottom-Up Parser LR Parser. Syntax-Directed Translation of Programming Language Constructs. Semantic Analysis: Type Checking. Principles of Code Generation.
<b>TEACHING FORMAT</b>	Frontal lectures and labs



## ASSESSMENT

Project: Compiler Development (30%)  
Final Written Exam (70%)  
The project will count for all 3 regular exam sessions.

## READING LIST

Textbook:

*Compilers: Principles, Techniques, and Tools*, Alfred V. Aho, Ravi Sethi and Jeff Ullman. Publisher: Prentice Hall, 2003.

Reading List:

*Compiler Construction: Principles and Practice*, Kenneth C. Loudon. 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

## LEARNING OUTCOME

As a final result of this Module, Students will be able to understand the Techniques used to generate a machine executable code starting from a source program. Furthermore, Students will be able to build such compilers and, in general, they will be able to write translator programs, i.e, programs that map whatever piece of source code into a new target code.

## COURSE PAGE

<http://www.inf.unibz.it/~artale/Compiler/compiler.htm>