Birla Institute of Technology & Science, Pilani
Hyderabad Campus

SECOND SEMESTER 2024-2025

Course Handout Part II

Date: 06-01-2025

In addition to part I (General Handout for all courses appended to the timetable), this portion gives further specific details regarding the course.

Course No.	: CS F363
Course Title	: Compiler Construction
Instructor-in-Charge	: Prof. Raghunath Reddy Madireddy
Instructors	: Prof. Chittaranjan Hota and Prof. Jabez Christopher

Scope and Objectives of the Course:

This course is an introductory course to compiler construction. In this course, students will learn the important basic elements of compilation to use the techniques effectively to design and build a working compiler. Topics include lexical analysis, parsing techniques, syntax-directed translation, symbol table, intermediate code generation, data flow analysis, code generation, code optimization, error detection, and recovery. Students will also participate in small teams to develop the building blocks of a compiler through a compiler project. This course also includes a lab to provide hands-on experience on tools for implementing a compiler using Lex/Flex, and Yacc.

- Gain an understanding of how compilers translate source code to machine executable code.
- Utilize tools to automate compiler construction.
- Comprehend how to perform parsing (top-down and bottom-up).
- Be familiar with techniques for simple code optimizations.
- Have the knowledge to design, implement, and test a compiler for a simple language, to include:
  - Implementing efficient mechanisms for lexical analysis.
  - Creating a parse table from a Context Free Grammar.
  - Implementing an efficient symbol table during the parsing phase.
  - Perform elementary semantic analysis checks on an abstract syntax tree.
  - Generating code for a target assembly language

Textbooks:

T1. **Aho, Sethi and Ullman**.**Compilers Principles, Techniques, and Tools.** Pearson Education. Low Price Edition. Second Edition, 2007.

Reference books:

R1. Andrew Appel, Modern Compiler Implementation in C. Cambridge University Press. (Foundation Books, New Delhi.) Rev. Ed. 2005.

R2. VINU V. DAS, Compiler Design Using FLEX and YACC, Prentice-Hall India

Course Plan:

Lecture No.	Learning objectives	Topics to be covered	Chapter in the Text Book
1	To understand the context and use of a compiler.	Introduction to Course. Structure and Components of a compiler.	T1 Ch1 (1.2)
2-4	To identify tokens and lexemes and to implement a lexer for a given context-free grammar	Tokens, Lexer functionality, and its implementation	T1 Ch. 3
	To list and identify various data structures that can be used in the implementation of the symbol table	Data Structures for Symbol Table Organization	T1 Ch 2 2.7
5-6	To understand CFG, parsing, and preprocessing grammars to parse by different parsing methods	Introduction to parsing	T1 Ch 4.2
		Some useful grammar transformations	T1 Ch 4.3
7-9	To able to understand top-down parsing	Recursive descent parser, LL(1) parser LL(1) Grammar LL(1) Parse algorithm Error recovery in LL parsing	T1 Ch 4.4
10-13	To be able to understand bottom-up parsing with and without look ahead symbols	Bottom Up parsers -LR(0),	T1 Ch 4.5, Ch 4.6
		More Powerful LR parsers: CLR(1) and LALR.	T1 Ch 4.7
14-15	To be able to handle ambiguous grammar and errors in LR parsing	Handling Ambiguous grammars	T1 Ch 4.8.1, 4.8.2
		Error recovery in LR parsing	T1 Ch 4.8.3
16-19	To be able to formulate their semantic grammar based on the task.	Inherited and Synthesized Attributes	T1 Ch. 5
20-24	To apply the knowledge of semantic grammar to generate 3AC for various programming language constructs like if statements, loops, functions, etc.	3AC, Syntax Trees, Translation of Expressions, Type Checking	T1 Ch. 6
25-26	To be able to perform optimization given a high-level language program.	Basic blocks, Flow graphs	T1 Ch. 8.4
		Optimization of basic blocks	T1 Ch. 8.5
		Loop optimizations	T1 Ch 9.1 and R1 Ch 18
		Global data flow analysis	T1 Ch. 8.5
27-28	To understand the implementation of the back end of a compiler - Code Generation and Register allocation.	Basic Blocks and Traces, Issues in code generation, Approach to code generation	T1 Ch. 8.6 and 8.8

Evaluation Scheme:

Component	Duration	Weightage (%)	Date & Time	Nature of Component
Mid-semester Exam	90 Mins	25%	08/03 9.30 - 11.00AM	Closed
Continuous Lab Evaluation	15 mins	7%	During the lab sessions (at least 4% before mid sem)	Open
Continuous Tutorial Evaluation	10 mins	7%	During the tutorial sessions (at least 4% before mid sem)	Open
Lab test	60 mins	10%	26 April 2025 (FN)	Closed
Project	Take home	16%	7% will be evaluated before mid and 9% after mid-sem	Open
Comprehensive Exam	180 Mins	35%	14/05FN	Closed

Mid-Semester grading: A minimum of 40% weightage will be considered for the mid-sem grading.

Chamber Consultation Hour: TBA

Notices: All notices related to the course will be displayed on LMS.

Make-up Policy: Make-up will be granted only to genuine cases with prior permission.

Academic Honesty and Integrity Policy: Academic honesty and integrity are to be maintained by all the students throughout the semester, and no academic dishonesty is acceptable.

INSTRUCTOR-IN-CHARGE
CS F363 Compiler Construction