3/4 B.Tech. FIRST SEMESTER

COMPILER DESIGN

CS5T4	**Required**	**Credits: 4**
Lecture: 4 periods/week		**Internal assessment: 30 marks**
Tutorial: 1 period /week		**Semester end examination: 70 marks**

Course context and Overview: The course is intended to teach the students the basic techniques that underlie the practice of Compiler Construction. The course will introduce the theory and tools that can be standardly employed in order to perform syntax-directed translation of a high-level programming language into an executable code. These techniques can also be employed in wider areas of application, whenever we need a syntax-directed analysis of symbolic expressions and languages and their translation into a lower-level description. They have multiple applications for man-machine interaction, including verification and program analysis.

Prerequisites: Formal Languages and Automata, Comparative Languages.

Objectives:

At the end of the course, students are expected to have:

1. An ability to use of formal attributed grammars for specifying the syntax and semantics of programming languages.
2. Working knowledge of the major phases of compilation, particularly lexical analysis, parsing, semantic analysis, and code generation.
3. An ability to design and implement a significant portion of a compiler for a language chosen by the instructor.

Learning Outcomes:

An ability to:

1). Understand about language processors and its phases.

2). Demonstrate about scanning of tokens and perform the syntax analysis by using parsing techniques.

3). Synthesize the parsers and select appropriate parsers for creation of new computer languages.

4). Perform Symantec analysis using attribute grammar and compare different memory management techniques in runtime environment.

5). Ascertain optimization techniques for intermediate code forms and code generation.

Unit I: Overview of language processing–preprocessors–compiler–assembler–interpreters – linkers & loaders - structure of a compiler – phases of a compiler.

Unit II: Lexical Analysis–Role of Lexical Analysis–Input Buffering–Specification ofTokens – Recognition of Token – The Lexical Analyzer Generator Lex.

Unit III: Syntax Analysis–Role of a parser–Context Free Grammar - Writing a Grammar

– Top Down Parsing – Recursive Descent Parsing – FIRST and FOLLOW – LL(1) Grammars – Non recursive Predictive Parsing – Error Recovery in Predictive Parsing.

Unit IV: Bottom up Parsing–Reductions–Handle Pruning - Shift Reduce Parsing - Introduction to simple LR – Why LR Parsers – Model of an LR Parsers –– Difference between LR and LL Parsers, Construction of SLR Tables.

Unit V: More powerful LR parses - Construction of CLR (1) - LALR Parsing tables, - Operator Precedence - Dangling ELSE Ambiguity - Error recovery in LR Parsing.

Unit VI: Semantic analysis–SDT -Evaluation Orders for SDD's- Construction of SyntaxTrees. Runtime Environment: Storage organization - Stack allocation - Access to non-local data - Heap management - Parameter passing mechanisms.

Unit VII: Intermediate code - DAG - Three address code–Quadruples - Triples - IndirectTriples – Basic Blocks – DAG representation of Block. Machine independent code optimization - Common sub expression elimination - Constant folding - Copy propagation - Dead code elimination - Strength reduction - Loop optimization -Procedure inlining.

Unit VIII: Machine dependent code optimization: Peephole optimization - Registerallocation - Instruction scheduling - Inter Procedural Optimization - Garbage collection via reference counting.

Learning Resources

Text Books:
1.Compilers: Principles, Techniques and Tools: 2nd Edition, Alfred V. Aho, Monica S. Lam, Ravi Sethi,Jeffrey D. Ulman; 2nd Edition ,Pearson Education.
2. Modern Compiler Implementation in C- Andrew N. Appel, Cambridge University Press.
Reference Books:

1. lex &yacc – John R. Levine, Tony Mason, Doug Brown, O'reilly
2. Modern Compiler Design- Dick Grune, Henry E. Bal, Cariel T. H. Jacobs, Wiley reamtech.
3. Engineering a Compiler-Cooper & Linda, Elsevier.
4. Compiler Construction, Louden, Thomson.
5. Principles of compiler design, V. Raghavan, 2nd ed, TMH, 2011.
6. http://www.nptel.iitm.ac.in/downloads/106108052/