AUTOMATA AND COMPILER DESIGN

III B.TECH - II SEMESTER

Course Code	Category	Hours/Week			Credits	Maximum Marks		
		L	T	P	C	CIE	SEE	Total
A6IT11	PCC	3	-	-	3	40	60	100

COURSE OBJECTIVES

The course should enable the students to:

1. Understand the fundamental concepts of formal languages and regular expressions.
2. Identify the need of compiler and stages of compiler.
3. Enumerating top down and bottom up parsing techniques used in compilation process.
4. Understand Syntax directed translation scheme and different ways of representing Intermediate code.
5. Issues in design of Code generation and different code optimization techniques.

COURSE OUTCOMES

The course should enable the students to:

1. Able to employ finite state machines and regular expressions for modelling and solving computing problems.
2. Classify compilation phases and construction of Derivation trees for the grammar.
3. Construct the parsing tables using different types of parsing approaches.
4. Able to apply Syntax directed translation for the given expression and different ways of representing Intermediate code.
5. Analyze the issues in code generation and applying different code optimization techniques.

UNIT - I	INTRODUCTION TO FINITE AUTOMATA	CLASSES: 13

Introduction to Finite Automata: Central Concepts of Finite Automata – Alphabets, Strings, Languages. Deterministic Finite Automata - Formal Definition, Design of DFA, Nondeterministic Finite Automata - Formal Definition, Design of NFA, Conversion from NFA with €-transitions to NFA without €-transitions. Conversion from NFA to DFA.

Regular Expressions: Introduction to Regular Expressions, Identity rules of Regular Expression, Algebraic Laws for Regular Expressions, Conversion of Finite Automata to Regular Expressions.

UNIT - II	INTRODUCTION TO COMPILERS	CLASSES: 12

Introduction to Compilers: Language Processing system, Phases of Compiler, Pass and Phases of Translation, Boot Strapping.

Lexical Analysis: Role of Lexical analyzer, Recognition of Tokens, Context Free Grammar, Derivations- Leftmost and Rightmost Derivations, Parse Trees, Ambiguity, Elimination of Left Recursion, Elimination of Left Factoring.

UNIT - III	TOP DOWN PARSING	CLASSES: 13

Top Down Parsing: Types of Parsers, Calculation of First and Follow, Construction of LL(1) Parsing table, Recursive Descent Parser.

Bottom up Parsing: Introduction, Classification of Bottom up parsing- LR Parser, Operator precedence Parser. Shift Reduce parser, Construction of parsing tables- LR(0), SLR(1), CLR(1), LALR(1).

UNIT - IV	SYNTAX-DIRECTED TRANSLATION	CLASSES: 10

Syntax-Directed Translation: Syntax-Directed Definition, Types of Attributes, Annotated parse tree, Syntax Directed Translation Scheme, SDT for Infix to Postfix.

Intermediate-Code Generation: Introduction, Types of Intermediate code, Implementation of Three Address Code.

UNIT-V	RUN-TIME ENVIRONMENTS	CLASSES: 12

Run-Time Environments: Source Language Issues, Storage Allocation Techniques, Activation Record.

Code Optimization: Principle sources of Code optimization, Basic Blocks, Optimization of Basic Blocks, Loop optimization, DAG representation of Basic Block.

Code Generation: Issues in design of Code Generation, Object code forms, Peephole Optimization.

TEXT BOOKS

1. Introduction to Automata Theory, Languages, and Computation, 3nd Edition, John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, Pearson Education.
2. Compilers: Principles, Techniques and Tools By Aho, Lam, Sethi, and Ullman, Second Edition, Pearson, 2014.
3. Aho, Sethi & Ullman, "Compilers: Principles, Techniques and Tools", Pearson Education

REFERENCE BOOKS

1. Compilers: Principles, Techniques and Tools By Aho, Sethi, and Ullman, Addison-Wesley, 1986
2. Introduction to Formal languages Automata Theory and Computation, Kamala Krithivasan, Rama R, Pearson.
3. V Raghvan, "Principles of Compiler Design", McGraw-Hill.