II B.Tech - II Semester
(17CS405) AUTOMATA AND COMPILER DESIGN

Int. Marks	Ext. Marks	Total Marks	L	T	P	C
40	60	100	3	1	-	3

Pre-Requisites: FLAT

UNIT-I: Formal Language and Regular Expressions: Languages, operations on languages, regular expressions (re), languages associated with (re), operations on (re), Identity rules for (re), Finite Automata: DFA, NFA, Conversion of regular expression to NFA, NFA to DFA. Applications of Finite Automata to lexical analysis, lex tools.

UNIT-II: Context Free grammars and parsing: Context free Grammars, Leftmost Derivations, Rightmost Derivations, Parse Trees, Ambiguity Grammars, Top-Down Parsing, Recursive Descent Parsers: LL(K) Parsers and LL(1)Parsers
Bottom up parsing: Rightmost Parsers: Shift Reduce Parser, Handles, Handle pruning, Creating LR (0) Parser, SLR (1) Parser, LR (1) & LALR (1) Parsers, Parser Hierarchy, Ambiguous Grammars, Yacc Programming Specifications.

UNIT-III: Syntax Directed Translation: Definitions, construction of Syntax Trees, S-attributed and L-attributed grammars, Intermediate code generation, abstract syntax tree, translation of simple statements and control flow statements

UNIT-IV: Semantic Analysis: Semantic Errors, Chomsky hierarchy of languages and recognizers, Type checking, type conversions, equivalence of type expressions, Polymorphic functions, overloading of functions and operators.

UNIT-V: Storage Organization: Storage language Issues, Storage Allocation, Storage Allocation Strategies, Scope, Access to Nonlocal Names, Parameter Passing, Dynamics Storage Allocation Techniques.

UNIT-VI: Code Optimization: Issues in the design of code optimization, Principal sources of optimization, optimization of basic blocks, Loop optimization, peephole optimization, flow graphs, Data flow analysis of flow graphs.
Code Generation: Issues in the design of code Generation, Machine Dependent Code Generation, object code forms, generic code generation algorithm, Register allocation and assignment, DAG representation of basic Blocks, Generating code from DAGs.

Course Outcomes:

CO-1	Analyze and design finite automata, and realize the use of Lexical analyzer.	L4
CO-2	Construct and identify the differences of top down and bottom-up parsers.	L3
CO-3	Syntax directed translation, synthesized and inherited attributes and Generate different types of intermediate codes.	L2
CO-4	Analyze and identify different types of formal languages, and grammars.	L4
CO-5	Design the good symbol table, to access easily.	L3
CO-6	Apply machine independent code optimization and code generation techniques.	L3

CO-PO/PSO Mapping Matrix:

	PO-1	PO-2	PO-3	PO-4	PO-5	PO-6	PO-7	PO-8	PO-9	PO-10	PO-11	PO-12	PSO-1	PSO-2	PSO-3
CO-1	2	1	2	1	3	-	-	-	-	-	-	1	1	-	-
CO-2	2	1	3	2	3	-	-	-	-	-	-	1	2	1	-
CO-3	2	1	3	2	3	-	-	-	-	-	-	1	2	1	-
CO-4	2	1	2	1	2	-	-	-	-	-	-	1	2	1	-
CO-5	2	1	3	1	1	-	-	-	-	-	-	1	1	1	-
CO-6	2	1	3	1	2	-	-	-	-	-	-	1	1	1	-

Text Books:

1. Introduction to Automata Theory Languages & Computation, 3/e, Hopcroft, Ullman, PEA
www.universityupdates.in || www.android.universityupdates.in www.universityupdates.in || www.android.universityupdates.in University Updates
2. Compilers Principles, Techniques and Tools, Aho, Ullman, Ravi Sethi, PEA

Reference Books:

1. Principles of Compiler Design, A.V. Aho . J.D.Ullman; PEA
2. Theory of Computer Science, Automata languages and computation , 2/e, Mishra, Chandra Shekaran, PHI
3. Elements of Compiler Design, A.Meduna, Auerbach Publications, Taylor and Francis Group.