II Year II Semester

L	T	P	C
3	1	0	3

Code: 17CS405

AUTOMATA AND COMPLIER DESIGN

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.

OUTCOMES:
- Classify machines by their power to recognize languages,
- Employ finite state machines to solve problems in computing,
- Explain deterministic and non-deterministic machines,
- Comprehend the hierarchy of problems arising in the computer science
- Acquire knowledge in different phases and passes of Compiler, and specifying different types of tokens by lexical analyzer, and also able to use the Compiler tools like LEX,YACC, etc.
- Parser and its types i.e. Top-down and Bottom-up parsers.

• Construction of LL, SLR, CLR and LALR parse table.
• Syntax directed translation, synthesized and inherited attributes.
• Techniques for code optimization.

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.