COMPILER DESIGN

Course Code:20CS1106

L	T	P	C
3	0	0	3

Course Outcomes :At the end of the Course the student will be able to:

CO1: Classify various phases of compiler.(L2)

CO2: Apply parsing techniques for a given grammar. (L3)

CO3: Illustrate various Syntax Directed Translation Schemes. (L3)

CO4: Model SDD's using Intermediate Representations. (L3)

CO5: Use algorithms to generate code for a target machine. (L3)

UNIT-I (10 Lectures)

INTRODUCTION TO COMPILING: Overview of Compilers, Phases of a Compiler, Bootstrapping.

LEXICAL ANALYSIS: The Role of Lexical Analyzer, Input Buffering, Specification of Tokens: Regular Expressions, Regular Definitions- Recognition of Tokens, A language for specifying Lexical Analyzers (LEX).

Learning Outcomes: At the end of the unit the student will be able to

1. understand the phases of a compiler.(L2)
2. classify the tokens and lexemes in a given input.(L2)
3. extend Regular Expressions for specifying tokens.(L2)

UNIT-II (10 Lectures)

SYNTAX ANALYSIS: The role of the Parser, Context free Grammars, Elimination of Left Recursion, Left factoring a grammar.

TOP-DOWN PARSING: Recursive descent Parsing, First and Follow, Predictive Parsing, LL (1) Grammars.

Learning Outcomes: At the end of the unit the student will be able to

1. construct Recursive Descent Parsing table for the given grammar (L3)
2. apply rules to make the grammar ready for parsing.(L3)
3. construct Predictive Parsing table for the given grammar(L3)

UNIT-III (10 Lectures)

BOTTOM-UP PARSING: Shift-Reduce Parser, LR Parsers SLR, Canonical LR, LALR, Operator Precedence Parser, Parser Generator (YACC).

SYNTAX-DIRECTED TRANSLATION: Syntax-DirectedDefinition, S-Attributed SDD, L-Attributed SDD, Translation Schemes.

LearningOutcomes: Attheendoftheunitthe studentwillbeable to

1. buildvariousLRParsingtablesforagivengrammar. (L3)
2. buildYACC Parser generatorforagivengrammar (L2)
3. compare S-Attributed SDD and L-Attributed SDD. (L2)

UNIT-IV (10 Lectures)

TYPE CHECKING: Type Systems, Specification of a Simple type checker, Equivalence of Type Expressions, Type Conversions.

RUN-TIME ENVIRONMENTS: Storage Allocation Strategies, Activation records, Access Links, Symbol Tables.

INTERMEDIATE CODE GENERATION: Intermediate Languages- Graphical Representations,Three Address Code,Implementations.

Learning Outcomes: At the end of the unit the student will be able to

1. build a type system for simple language.(L3)
2. explain the two kinds of type conversions.(L2)
3. summarize various storage allocation strategies.(L2)
4. develop various representations for three address code.(L3)

UNIT- V (10 Lectures)

CODE OPTIMIZATION: Introduction, Principle sources of optimization.

CODE GENERATION: Issues in the Design of a Code Generator, The Target Language, Basic Blocks and Flow Graphs, Peephole optimization.

Learning Outcomes: At the end of the unit the student will be able to

to1.applyoptimizationtechniquesonagivencode.(L3)

1. apply optimization techniques on a given code.(L3)
2. build a flow graph for the identified basic blocks. (L3)
3. apply rules to design a simple code generator.(L3)

TEXTBOOKS:

1. Alfred Aho, Monica S Lam, Ravi Sethi, JeffreyD.Ullman, *Compilers- PrinciplesTechniques and Tool,* 2nd Edition, Pearson Education India, 2013.

REFERENCE BOOKS:

1. V. Raghavan, *Principles of Compiler Design*,1st Edition, McGraw Hill Education, 2017.
2. Alfred V Aho, Ravi Sethi, Jeffrey D.Ullman,*Compilers- PrinciplesTechniques,andTool,* 2nd Edition, Pearson Education, 2013.
3. Kenneth C.Louden, *Compiler Construction Design*, 2nd Edition, Cengage, 2010.

WEB REFERENCES:

1.https://swayam.gov.in/nd1_noc20_cs13/preview