COMPILER DESIGN
(Common to CSE& IT)

Course Code :13CT1123

L	T	P	C
4	1	0	3

Course Educational Objectives:

The main objective of the course is to give an overall idea about the compiler development process. Upon completion of this course the student should be able to:

- Analyze the source code and differentiate between lexical, syntax and semantic errors.
- Understand the run time storage requirements to run a program.
- Optimize the source code by applying optimization techniques.
- Develop a Compiler by having an idea of the six different phases.
- Giving idea about the Data-Flow Analysis of Structured Flow graphs

Course Outcomes:

At the end of the course the student will be able to

- Understand the internal process of Compilation
- Understand Lexical Analyzer
- Understands both top-down and bottom-up parsers
- Understand Semantic Analyzer
- Understands intermediate code generation and optimization techniques

UNIT-I (12 Lectures)

INTRODUCTION TO COMPILING:

Overview of Compilers, Analysis of the Source Program, the Phases of a Complier, Pre-Processors, Assemblers, Two Pass Assembly, Loaders and Link-Editors, Bootstrapping, The Grouping of Phases, Compiler Construction Tools.

UNIT-II (12 Lectures)

LEXICAL ANALYSIS:

The Role of the Lexical Analyzer, Strings and Languages, Operations on Languages, Regular Expressions, Regular Definitions, Notational Shorthands, Recognition of Tokens, A Language for specifying Lexical Analyzers(LEX).

SYNTAX ANALYSIS:

The Role of the Parser, Context-free Grammars, Writing a Grammar.

UNIT-III (12 Lectures)

TOP-DOWN PARSING:

Recursive Descent Parsing, Predictive Parsers, Non-Recursive Predictive Parsing, First and Follow, Construction of Predictive Parsing Tables, LL(1) Grammars, Error Recovery in Predictive Parsing.

BOTTOM-UP PARSING:

Handles, Handle Pruning, Stack Implementation, Operator-Precedence Parsing, LR Parsers-SLR, Canonical LR, LALR. Using Ambiguous Grammars, Parser Generator (YACC).

SYNTAX-DIRECTED TRANSLATION:

Syntax-Directed Definition, Construction of Syntax Trees, S-Attributed Definitions, L-Attributed Definitions.

UNIT-IV (12 Lectures)

SEMANTIC ANALYSIS:

Type Systems, Specification of a Type Checker, Equivalence of type-expressions, Type Conversions, Overloading of functions and operators, Polymorphic functions, Algorithm for Unification.

RUN-TIME ENVIRONMENT:

Source Language Issues, Storage Organization, Storage Allocation Strategies, Blocks, Access Links, Procedure Parameters, Displays, Parameter Passing, Symbol Tables.

UNIT-V (12 Lectures)

INTERMEDIATE CODE GENERATION:

Intermediate Languages-Graphical Representations, Three Address Code, Implementations, Boolean Expressions.

CODE OPTIMIZATION:

Introduction, Principle sources of Optimization, Optimization of Basic Blocks.

CODE GENERATION:

Issues, the Target Machine, Run-Time Storage Management, Basic Blocks and Flow graphs, Loops in Flow graphs, Data-Flow Analysis of Structured Flow graphs, Peephole Optimization, DAG, Simple Code Generator.

TEXT BOOKS:

1. Alfred V Aho, Ravi Sethi, Jeffrey D.Ullman, Compilers-Principles Techniques and Tools, 2nd Edition, Pearson Education, 2008.

REFERENCES:

1. Raghavan, *"Principles of Compiler Design"*, 2nd Edition, TMH, 2011.
2. Kenneth C.Louden, *"Compiler Construction-Principles and Practice"*, 2nd Edition, Cengage, 2010.
3. Cooper and Linda, *"Engineering a Compiler"*, 4th Edition, Elsevier, 2008.

WEB REFERENCES:

http://nptel.iitm.ac.in/courses/Webcourse-contents/IIT-%20Guwahati/afl/index.htm