COURSE CODE	COURSE TITLE	L	T	P	C
1151IT112	COMPILER DESIGN	3	0	0	3

Course Category: Program Core

A. Preamble :

This Course describes the theory and practice of compilation, in particular, the lexical analysis, parsing and code generation and optimization phases of compilation, and design a compiler for a concise programming language.

B. Prerequisite Courses:

Sl. No	Course Code	Course Name
1	1150CS201	Problem Solving using C
2	1150MA202	Mathematics II

C. Related Courses:

Sl. No	Course Code	Course Name
1	1156IT601	Minor Project
2	1156IT701	Major Project

D. Course Outcomes :

Upon the successful completion of the course, students will be able to:

CO No's	Course Outcomes	Knowledge Level (Based on revised Bloom's Taxonomy)
CO1	Use the knowledge of patterns, tokens & regular expressions for solving a problem	K2
CO2	Apply the knowledge of Lex tool & YAAC tool to develop scanner & parser	K3
CO3	Explain a intermediate code generator	K3
CO4	Design & conduct experiments for intermediate generation in compiler	K3
CO5	Learn the new code optimization technique to improve the performance of a program in terms of speed and space	K3

E. Correlation of COs with POs :

COs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	**H**											
CO2	**H**	**M**										
CO3		**M**										
CO4	**M**											
CO5	**H**	**M**										

H- High; M-Medium; L-Low

F. Course Content:

UNIT I Introduction to Compilers L – 9

Compilers, Analysis of the Source Program, The Phases of a Compiler, Cousins of the Compiler, The Grouping of Phases, Compiler-Construction Tools. LEXICAL ANALYSIS: Need and role of lexical analyzer-Lexical errors, Input Buffering - Specification of Tokens, Recognition of Tokens, Design of a Lexical Analyzer Generator

UNIT II Syntax Analysis L – 9

Need and role of the parser- Context Free Grammars-Top Down parsing - Recursive Descent Parser - Predictive Parser - LL(1) Parser -Shift Reduce Parser - LR Parser - LR (0) item - Construction of SLR Parsing table -Introduction to LALR Parser, YACC- Design of a syntax analyzer for a sample language

UNIT III Intermediate Code Generation L – 9

Intermediate languages – Declarations – Assignment Statements – Boolean Expressions – Case Statements – Back patching – Procedure calls.

UNIT IV Code Generation L – 9

Issues in the design of code generator – The target machine – Runtime Storage management – Basic Blocks and Flow Graphs – Next-use Information – A simple Code generator – DAG representation of Basic Blocks

UNIT V Code Optimization and Run Time Environments L – 9

Introduction– Principal Sources of Optimization – Peephole Optimization- Optimization of basic Blocks – Introduction to Global Data Flow Analysis – Runtime Environments – Source Language issues – Storage Organization – Storage Allocation strategies – Access to non-local names – Parameter Passing**.**

TOTAL : 45 Periods

G. Learning Resources

i. Text Books:

1.Alfred Aho, Ravi Sethi, Jeffrey D Ullman, "Compilers Principles, Techniques and Tools", Pearson Education Asia, 2003.

i. Reference Books:

1. Allen I. Holub "Compiler Design in C", Prentice Hall of India, 2003.
2. C. N. Fischer and R. J. LeBlanc, "Crafting a compiler with C", Benjamin Cummings, 2003.
3. J.P. Bennet, "Introduction to Compiler Techniques", Second Edition, Tata McGraw-Hill, 2003.
4. Henk Alblas and Albert Nymeyer, "Practice and Principles of Compiler Building with C", PHI, 2001.
5. Kenneth C. Louden, "Compiler Construction: Principles and Practice", Thompson Learning, 2003

ii. Online Recourses:

1. http://www.tutorialspoint.com/compiler_design/
2. http://nptel.ac.in/courses/106104123/Compiler%20DesignQuestions.pdf
3. http://www.vssut.ac.in/lecture_notes/lecture1422914957.pdf