

Branch Name:	Computer Engineering
Semester/Year:	Semester VII / Final Year
Subject Title:	Compiler Design
Subject Code:	1ET1030703
Pre-requisite:	Basic idea regarding formal languages and good knowledge of data structures & algorithms

Course Objective:

- Learn the design principles of a Compiler.
- Learn the various parsing techniques and different levels of translation.
- Learn how to optimize and effectively generate machine codes
- To introduce LEX and YACC tools.
- To learn to develop algorithms to generate code for a target machine.

Teaching Scheme (Hours per week)				Evaluation Scheme (Marks)				
Lecture (L)	Tutorial (T)	Practical (P)	Credit	Theory (Marks)		Practical (Marks)		Total (Marks)
				University Assessment	Continuous Assessment	University Assessment	Continuous Assessment	
04	-	02	05	70	30	30	20	150

Subject Contents			
Sr. No	Topic	Total Hours	Weightage (%)
1	INTRODUCTION TO COMPILER Basics of Compiler, Difference between interpreter, assembler and compiler, Overview and use of linker and loader, Types of Compilers, Analysis of the Source Program , Components of compiler , Errors encountered in different phases of compiler , Cousins of the compiler , The grouping of phases , Front end and Back end of compiler, Pass structure	6	08%
2	LEXICAL ANALYSIS Introduction to Lexical Analyzer, Input Buffering, Specification of Tokens, Recognition of Tokens , Error Detection & Recovery , A Language for Specifying Lexical Analyzers , Finite Automata From a Regular Expression , Design of a Lexical Analyzer Generator , Optimization of DFA , Lexical analyzer generator: LEX	8	15%

3	SYNTAX ANALYSIS The Role of the Parser, Types of grammar, CFG, Left most derivation , Right most derivation, Parse Tree, Restriction on CFG, Ambiguous grammar, Top-Down Parsing, Issues of CFG, Recursive Decent Parser , Construction of Predictive Parsing Table , LL (1) Grammar, String Parsing using M-Table, Bottom-Up Parsing: Handle, Shift-Reduce parser, LR parsers: LR (0), SLR (1), LALR (1), CLR (1), String parsing procedure using LR parser, R-R and S-R Conflicts , Operator - Precedence Parsing ,Parser Generator :YACC	9	22%
4	SYNTAX DIRECTED TRANSLATION Syntax - Directed Definitions, Construction of Syntax Trees, Bottom – Up evaluation of S -Attributed Definitions, L-Attributed Definitions, Translation schemes	4	10%
5	COMPILE-TIME ERROR HANDLING Overview of Error Handling ,Error Detection ,Error Recovery ,Error Repair	4	8%
6	INTERMEDIATE CODE GENERATION Different Intermediate Forms, Syntax Directed Translation Mechanisms, Attributed Mechanisms And Attributed Definition	4	9%
7	RUNTIME ENVIRONMENT Source Language Issues , Storage Organization , Storage - Allocation Strategies , Access to Non local Names , Parameter Passing , Symbol Tables, Dynamic Storage Allocation Techniques	5	10%
8	CODE OPTIMIZATION Global Data Flow Analysis , A Few Selected Optimizations like Command Sub Expression Removal , Loop Invariant Code Motion , Strength Reduction etc	4	9%
9	CODE GENERATION Issues in the Design of a Code Generator , The Target Machine , Run-Time Storage Management , Basic Blocks and Flow Graphs , Next - Use Information , A Simple Code Generator , Register Allocation and Assignment , The DAG representation of Basic Blocks , Peephole Optimization, Generating Code from DAGs	4	9%

Course Outcome:

At the end of the course, the student should be able to:

- Design and implement a prototype compiler.
- Use the different compiler construction tools.
- Able to design and implement LL and LR parsers.
- Able to design algorithms to perform code optimization in order to improve the performance of a program in terms of space and time complexity.

List of Text Books:

1. Compilers: Principles, Techniques and Tools By Aho, Lam, Sethi, and Ullman, Second Edition, Pearson, 2014.

List of Reference Books:

1. Steven S. Muchnick , “Advanced Compiler Design and Implementation, “Morgan Kaufmann Publishers – Elsevier Science, India, Indian Reprint 2003
2. The Theory and Practice of Compiler writing, J. P. Tremblay and P. G. Sorenson, TMH
3. Compiler Construction-Principles and Practice, Kenneth C Loudon, Cengage Learning.

List of Suggested titles of Experiments:

1. To Study about Lexical Analyzer Generator(LEX) and Flex(Fast Lexical Analyzer)
2. The program replaces the substring abc by ABC from the given input string.
3. Write a LEX program to check Well formedness of brackets
4. Write a LEX program to count number of vowels and consonant in a given string.
5. Write a LEX program to find the capital letters in given string.
6. Write a LEX Program to count the number of characters, words, spaces and Lines in a given input file.
7. Write a program in C to implementing Symbol Table.
8. Implement lexical analyzer using lexical analyzer generating tools.
9. Write a LEX program to convert decimal number into hexadecimal.
10. To Study about Parser Generator Tool :YACC
11. To write a program for implementing a calculator for computing the given expression using semantic rules of the YACC tool.
12. Implement using YACC a Program to recognize a valid arithmetic expression that uses operators +, -, * and /.