College of Engineering, Pune - 05
Department of Computer Engineering and Information Technology
COMPILER CONSTRUCTION
Branch: Computer Science & Engineering
Teaching Scheme: Lectures- 3 Hrs/Week **Examination Scheme: 100 marks**
Continuous evaluation-
Semester: VII Teacher's Assessment – 20 marks
Mid Sem Exam- 30 marks
End Sem Exam - 50 marks

Teaching plan

No.	Unit	Topic	Count	Total
1	I	Introduction: translator issues, Translator Issues, why to write a Compiler,	1	4
		what is a Compiler, what is the Challenge ,Compiler Architecture,	1	
		front end and Back end model of compiler,	1	
		Incremental compiler, Boot strapping.	1	
2	II	Lexical Analysis: Concept of Lexical Analysis, Regular Expressions, Deterministic finite automata (DFA),	1	6
		Non- Deterministic finite automata (NFA), Converting regular expressions to DFA, Converting NFA to DFA	2	
		Hand coding of Lexical analyzer,	1	
		Introduction to LEX Tool and LEX file specification,	1	
		Error detection and recovery in LEX.	1	
3	III	Syntax Analysis: Context Free Grammars(CFG),Concept of parsing, Parsing Techniques	1	8
		Top-Down Parsers : Introduction, Predictive Parsing - Removal of left recursion, Removal of left factoring,	1	
		Recursive Descent Parsing, Predictive LL(k) Parsing Using Tables, Bottom Up parsing	1	
		Introduction, Shift-Reduce Parsing Using the ACTION/GOTO Tables, Table Construction, SLR(1),	1	
		LR(1), and LALR(1) Grammars,	2	
		Practical Considerations for LALR(1) Grammars, Introduction to YACC Tool	1	
		YACC file specification, Error detection and recovery in YACC.	1	
4	IV	Semantic Analysis & Intermediate Representation :Need of semantic analysis, Abstract Parse trees for Expressions, variables, statements, functions and class declarations,	1	8
		Syntax directed definitions, Syntax directed translation schemes for declaration processing, type analysis, scope analysis ,	1	
		Symbol Tables (ST),Organization of ST for block structure and non block structured languages, Symbol Table management,	1	
		Type Checkers : type checking for expressions, declarations (variable, type, function, recursive), statements,	1	
		Intermediate code generation: Intermediate languages, Design issues, Intermediate representations: three address, postfix	2	
		& abstract syntax trees, Intermediate code generation for declaration, assignment, iterative statements, case statements, arrays	1	
		Structures, conditional statements, Boolean expressions, procedure/	1	

		function definition and call.		
5	V	Run-Time Memory Management & Code generation: Model of a program in execution, Stack and static allocation,	1	7
		Activation records , Issues in the design of code generation,	2	
		Target machine description, Basic blocks & flow graphs, Expression Trees,	1	
		Unified algorithms for instruction selection and code generation.	1	
		Sethi Ullman algorithm for expression trees , Aho Johnson algorithm, Different models of machines	1	
		order of evaluation, register allocation , Code generator-generator concept.	1	
6	VI	Code Optimization Introduction, Principal sources of optimization,	1	7
		Machine Independent Optimization, Machine dependent Optimization	1	
		Various Optimizations: Function preserving transformation, Common Sub-expressions	2	
		Copy propagation, Dead-code elimination, Loop Optimizations,	1	
		reduction in strength, Peephole Optimization,	1	
		Code Motion, Induction variables & Redundant –instruction elimination	1	
			total	40

Text Books :

- Alfred V. Aho, Monica S.Lam, R. Sethi and J.D. Ullman “Compilers: principles, techniques and tools” Pearson Education, 2011.

References :

1. Andrew W Appel, Modern Compiler Implementation in C, Cambridge University Press; 1997
2. Kenneth C Louden, “Compiler Construction Principle and Practice”, PWS publishing Company, 1997
3. Dhamdhere D.M., “Compiler Construction Principle and Practice”, Mac. Millan India, New Delhi,1983
4. Holub, A.J., “Compiler design in C” –Prentice Hall,1982
5. John Levine, Tony Mason & Doug Brown, “Lex and Yacc”, O’Reilly.1995

Useful URLs:

1. The lex and Yacc page : http://dinosaur.compilertools.net/
2. http://www.personal.kent.edu/~rmuhamma/Compilers/compiler.html
3. http://www.cs.sjsu.edu/~louden/cmptext/
4. http://www.gtoal.com/software/CompilersOneOhOne/
5. http://lambda.uta.edu/cse5317/notes/node4.html

Course Outcomes:

Students will be able to:

1. Demonstrate the understanding of different phases of compilation.
2. Demonstrate the ability to generate and code lexical and syntax analyzer.
3. Analyze and differentiate different parsing techniques and syntax directed translation schemes and choose the optimal parsing technique.
4. Apply different intermediate code generation representation for program construct.
5. Identify different code optimization techniques for the various statements

(Dr.S.N. Ghosh)
Subject In charge

Head
Dept. of Computer Science & Engg.

COMPILER CONSTRUCTION LABORATORY	
Teaching Scheme Practical : 3 hrs/week	Examination Scheme Practical –50 marks Term Work – 50 marks

	List of Assignments:
1	Calculator (text or graphics) using LEX and YACC or Document Editor (find, replace, macro) using LEX and YACC, or Similar kind of assignment using LEX and YACC.
2	Lexical Analyzer for extracting noun and verb phrases from the input English text document.
3	Syntax Analyzer along with Intermediate code generation (Triple, Quad) for a subset of English language.
4	Any two optimization techniques on Intermediate Code Generation Constant expression evaluation. Local copy propagation. Common sub expression elimination. Loop invariant code movement.

Course Outcomes:
Students will be able to:
1. Implement lexical analyzers using Lex tool
2. Write a parser and semantic analyzer for different Context-Free Grammars using Yacc tool.
3. Implement different representations of Intermediate code
4. Demonstrate ability to optimize intermediate code using different techniques

(Dr. S.N. Ghosh)
Subject In charge

Head
Dept. of Computer Science & Engg.