`

National Institute of Technology Meghalaya

An Institute of National Importance

CURRICULUM

Programme	**Master of Computer Applications**	Year of Regulation	**2024-25**
Department	**Computer Science and Engineering**	Semester	**IV**

Course Code	Course Name	Pre-Requisite	Credit Structure				Marks Distribution			
			L	T	P	C	INT	MID	END	Total
CA504	**Compiler Design**		**3**	**0**	**0**	**3**	**50**	**50**	**100**	**200**

			CO's	**Statement**	**Bloom's Taxonomy**
Course Objectives	The Objectives of this course is to explore the principles, algorithms, and data structures involved in the design and construction of compilers.	Course Outcomes	CA504.1	Able to acquire knowledge about lexical, syntactic and semantic structures of any computer programming language.	Understand
	To discuss context-free grammars, and front-end phases of a compiler: lexical analysis, parsing techniques, symbol tables, error recovery.		CA504.2	Able to analyse and separate the lexical, syntactic and semantic analysis into meaningful phases for a compiler to undertake language translation.	Analyse
	To discuss back-end phases of a compiler: code generation, and different code optimization techniques.		CA504.3	Able to design Write a scanner, parser, and semantic analyser for limited form of C like programming languages.	Create
			CA504.4	Able to convert source code in simple language into machine code for a novel computer.	Create
			CA504.5	Able to describe techniques for intermediate code and machine code optimisation.	Create
			CA504.6	Able to design the structures and support required for compiling advanced language features.	Create

COs	Mapping with Program Outcomes (POs)												Mapping with PSOs		
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CA504.1	**3**	**2**	**3**	**1**									**1**	**2**	**2**
CA504.2	**3**	**3**	**3**	**3**								**2**	**1**	**1**	**3**
CA504.3	**2**	**3**	**3**	**1**	**3**				**1**				**1**	**1**	**3**
CA504.4	**2**	**1**	**1**	**2**	**2**				**1**				**1**	**1**	**3**
CA504.5	**2**	**1**	**2**	**1**	**1**								**1**	**1**	**3**
CA504.6	**2**	**2**	**2**	**3**								**2**	**1**	**1**	**3**
CA504	**2.3**	**2**	**2.3**	**1.8**	**2.0**				**2**			**2**	**1**	**1.2**	**2.8**

SYLLABUS

No.	Content	Hours	COs
I	Introduction to Compiler, Phases and passes,	**02**	**CA504.1**
II	Finite state machines and regular expressions and their applications to lexical analysis, Implementation of lexical analyzers, lexical-analyzer generator, LEX-compiler: LEX/FLEX,	**06**	**CA504.1, CA504.2, CA504.3**
III	Formal grammars and their application to syntax analysis, BNF notation, ambiguity, YACC. The syntactic specification of programming languages: Context free grammars, derivation and parse trees, capabilities of CFG. Basic Parsing Techniques: Parsers, Shift reduce parsing, operator precedence parsing, top down parsing, predictive parsers Construction of efficient Parsers: LR parsers, the canonical Collection of LR(0) items, Constructing SLR parsing tables, constructing Canonical LR parsing tables, Constructing LALR parsing tables, Using ambiguous grammars, an automatic parser generator, implementation of LR parsing tables, constructing LALR sets of items.	**14**	**CA504.1, CA504.3**
IV	Syntax-directed Translation: Syntax-directed Translation schemes, Implementation of Syntax directed Translators, Intermediate code, postfix notation, Parse trees & syntax trees, three address code, quadruple & triples, Translation of assignment statements, Boolean expressions, statements that alter the flow of control,	**11**	**CA504.4,CA504.5**

	Postfix translation, translation with a top down parser. More about translation: Array references in arithmetic expressions, procedures call, declarations, case statements. Symbol Tables: Data structure for symbols tables, representing scope information.		
V	Run-Time Administration: Implementation of simple stack allocation scheme, Storage allocation in block structured language. Error Detection & Recovery: Lexical Phase errors, syntactic phase errors semantic errors. Introduction to code optimization: Loop optimization, DAG representation of basic blocks, Value numbers and algebraic laws, Global Data-Flow analysis.	**09**	**CA504.1, CA504.6**
Total Hours		**42**	

Essential Readings

1. A.V. Aho, M. S. Lam, R. Sethi and J. D. Ullman, “Compilers-Principles, Techniques and Tools”, 2nd ed., 2007, Pearson Education.
2. K. Muneeswaran, “Compiler Design”, 1st ed., 2013, Oxford Publication.
3. P.H. Dave, H.B. Dave, “Compilers: Principles and Practice”, 1st ed. 2012, Pearson Education.

Supplementary Readings

1. Allen I. Holub, “Compiler Design in C”, 1st ed.(Indian print), 2012, PHI.
2. John Levine, “Flex & Bison “, 1st ed., 2009, O’reilly.
3. Torben Ægidius Mogensen, “Basics of Compiler Design”, 1st ed., 2007, DIKU, University of Copenhagen