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	Continuous Evaluation	Quiz/ Viva	Total
CA554	**Compiler Design Lab**		**0**	**1**	**2**	**2**	**70**	**30**	**100**

Course Objectives		Course Outcomes	**CO's**	**Statement**	**Bloom's Taxonomy**
	The Objectives of this course is to explore the principles, algorithms, and data structures involved in the design and construction of compilers.		CA554.1	Specify and analyse the lexical, syntactic and semantic structures of any computer programming language.	Analyse
	To implement some phases of the front-end of a general compiler.		CA554.2	Separate the lexical, syntactic and semantic analysis into meaningful phases for a compiler to undertake language translation.	Create
	To implement some phases of the backt-end of a general compiler		CA554.3	Write a scanner, parser, and semantic analyser for limited form of C like programming languages.	Create
			CA554.4	Convert source code in simple language into machine code for a novel computer.	Create
			CA554.5	Describe techniques for intermediate code and machine code optimisation.	Understand

COs	Mapping with Program Outcomes (POs)												Mapping with PSOs		
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CA554.1	**3**	**2**	**3**	**1**									1	2	2
CA554.2	**3**	**3**	**3**	**3**								2	1	1	3
CA554.3	**2**	**3**	**3**	**1**	**3**				1				1	1	3
CA554.4	**2**	**1**	**1**	**2**	**2**				1				1	1	3
CA554.5	**2**	**1**	**2**	**1**	**1**								1	1	3
CA554	2.40	2.00	2.40	1.60	2.00				1.00			2.00	1.00	1.20	2.80

SYLLABUS

No.	Content	Hours	COs
I	1) Using Lex/Flex , write a program to append line number before each (i) lines(empty/non-empty). (ii) non-empty lines Input/output streams may be files. 2) Using Lex/Flex , write a program to count number of lines, words, visible characters, total characters. Input/output streams may be files.	**4**	**CA554.1, CA554.2, CA554.3**
II	3) Using Lex/Flex , write a program to identify some keywords, identifiers, integers and real numbers from a simple C program. Input/output streams may be files. 4) Lex program to copy a file by replacing multiple sequences of white spaces with a single white space. [blanks/tab => blank, more than one " \n" => " \n"]. 5) Also add removal of comments in above program.	**2**	**CA554.1, CA554.2, CA554.3**
III	6) Lex program to copy a C program by replacing each instance of the keyword *float* by *double*. 7) Write a Lex program that converts a file to "Pig Latin". Specifically, assume the file is sequence of English words (group of letters) separated by white space. Every time a word is encountered: 1. If the first letter is consonant, move it to the end of the word and then add ay. 2. If the first letter is a vowel, just add ay to the end of the word.	**2**	**CA554.1, CA554.2, CA554.3**
IV	8) Using Lex/Flex , write a program to encode and decode.	**2**	**CA554.1, CA554.2, CA554.3**
V	9) Using Lex/Flex , write a program to (i) identify the Roman numbers (ii) add 2 Roman numbers.	**2**	**CA554.1, CA554.2, CA554.3**
VI	10) Create a recursive predictive parser for a grammar(as given in lab class).	**2**	**CA554.1, CA554.2, CA554.3**
VII	11) Create a non-recursive predictive parser(LL parser) for a grammar(as given in lab class).	**2**	**CA554.1, CA554.2, CA554.3**

VIII	12) Using Flex and Bison tools, create a calculator program that support addition,subtraction, multiplication, division, power operations on numbers and variables.	**4**	**CA554.1, CA554.2, CA554.3**
IX	13) Using Flex and Bison tools, create a translator to convert a simple program written in arbitrary language to a program in C language.	**4**	**CA554.1,CA554.4**

X	14) Using Flex and Bison tools, create a program to convert a simple assignment expression into intermediate code.
	Ex:- input: z = -(a+b-c)
	output:
	t1 = a + b
	t2 = t1 – c
	t3 = - t2
	z = t3
		4	**CA554.1,CA554.5**

Total Hours		**28**	

Essential Readings

1. A.V. Aho, M. S. Lam, R. Sethi and J. D. Ullman, “Compilers-Principles, Techniques and Tools”, 2nd ed., 2006, 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