Code No.: R22CS502PC | R22 | H.T.No. | | | 8 | R | | | | | |

CMR ENGINEERING COLLEGE: : HYDERABAD
UGC AUTONOMOUS
III–B.TECH–I–Semester End Examinations (Regular) - December- 2025
AUTOMATA THEORY AND COMPILER DESIGN
(Common for CSE, CSM, CSC)

[Time: 3 Hours] **[Max. Marks: 60]**

Note: This question paper contains two parts A and B.
Part A is compulsory which carries 10 marks. Answer all questions in Part A.
Part B consists of 5 Units. Answer any one full question from each unit. Each question carries 10 marks.

PART-A **(10 Marks)**

1.	a)	Define Finite Automata.	[1M]
	b)	Distinguish between Deterministic and Non-deterministic finite automata.	[1M]
	c)	List out the algebraic laws in optimizing pattern recognition expressions.	[1M]
	d)	Assess whether leftmost and rightmost derivations always produce the same parse tree for an unambiguous grammar.	[1M]
	e)	Analyze the equivalence between PDA and CFG.	[1M]
	f)	Write an undecidable problem related to Turing Machines.	[1M]
	g)	Illustrate how input buffering improves the efficiency of the lexical analyzer.	[1M]
	h)	Construct a grammar to generate simple arithmetic expressions containing the operators + and *.	[1M]
	i)	Write a simple syntax-directed definition for arithmetic expressions.	[1M]
	j)	Define stack allocation.	[1M]

PART-B **(50 Marks)**

2.		Explain the importance of finite automata in understanding computational complexity and problem-solving.	[10M]
		OR	
3.		Design an NFA for strings over {0,1} that end with "01" and convert it into an equivalent DFA.	[10M]
4.	a)	Construct Finite Automata for the given regular expression a(a+b)*b	[5M]
	b)	Prove that the language L= $\{a^n b^n / n>=1\}$ is not regular using Pumping Lemma.	[5M]
		OR	
5.		Design a CFG for arithmetic expressions involving +, *, and parentheses, and generate its parse tree for a given input.	[10M]
6.		Design a Turing Machine to accept the Language L=$\{WCW^R / W \in (a+b)^*\}$.	[10M]
		OR	
7.		Analyze the relationship between recursively enumerable languages and Turing Machines.	[10M]
8.		Explain the role of Lexical analyzer to recognize numbers, operators and identifiers in a given source code.	[10M]
		OR	
9.		Construct the parse tree and leftmost derivation for the string id + id * id using a suitable grammar.	[10M]
10.		Explain the evaluation orders for SDDs.	[10M]
		OR	
11.		Generate the three-address code for the expression (a + b) * (c - d).	[10M]

****

$\in$