



SIDDHARTH INSTITUTE OF ENGINEERING & TECHNOLOGY:PUTTUR (AUTONOMOUS)

Siddharth Nagar, Narayanavanam Road – 517583

QUESTION BANK (DESCRIPTIVE)

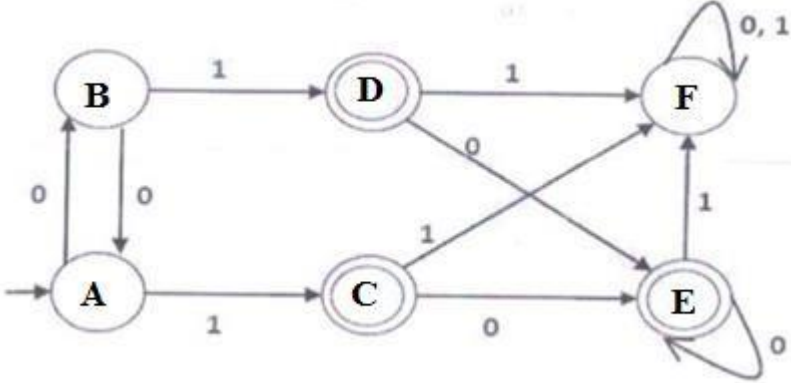
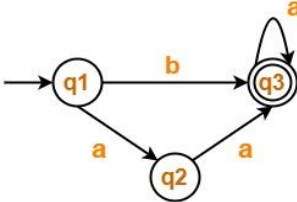
Subject with Code: Automata Theory and Compiler Design (23CS0518)

Course & Branch: III B.Tech – CSE Year & Sem: III &I Regulations: R23

UNIT –I

INTRODUCTION TO AUTOMATA AND REGULAR EXPRESSIONS

1	a	Define alphabet & strings.	[L1][CO1]	[2M]																	
	b	State Arden's theorem.	[L1][CO1]	[2M]																	
	c	Define Grammar.	[L1][CO1]	[2M]																	
	d	Write the mathematical definition of finite automata.	[L1][CO1]	[2M]																	
	e	What is Regular Expression and regular set.	[L1][CO1]	[2M]																	
2	a	Describe String acceptance and check whether the given strings accepted by FA or not. <table border="1"><thead><tr><th rowspan="2">States (Q)</th><th colspan="2">Input Alphabtes</th></tr><tr><th>0</th><th>1</th></tr></thead><tbody><tr><td>→q0</td><td>q1</td><td>q3</td></tr><tr><td>q1</td><td>q0</td><td>q2</td></tr><tr><td>q2</td><td>q3</td><td>q1</td></tr><tr><td>q3</td><td>q2</td><td>q0</td></tr></tbody></table> (i) 0001 (ii) 1010 (iii) 101101	States (Q)	Input Alphabtes		0	1	→q0	q1	q3	q1	q0	q2	q2	q3	q1	q3	q2	q0	[L2][CO1]	[5M]
States (Q)	Input Alphabtes																				
	0	1																			
→q0	q1	q3																			
q1	q0	q2																			
q2	q3	q1																			
q3	q2	q0																			
	b	Analyze Chomsky Hierarchy of Languages	[L4][CO1]	[5M]																	
3	a	Compare DFA and NFA	[L4][CO1]	[5M]																	
	b	Design DFA which accepts even number of 0's and odd number of 1's over {0, 1}.	[L6][CO1]	[5M]																	
4		Describe the procedure of conversion of NFA to DFA. Convert the given NFA to DFA. <table border="1"><thead><tr><th rowspan="2"></th><th colspan="2">Next state</th></tr><tr><th>0</th><th>1</th></tr></thead><tbody><tr><td>→ q0</td><td>q0,q1</td><td>q0</td></tr><tr><td>q1</td><td>q2</td><td>q1</td></tr><tr><td>q2</td><td>q3</td><td>q3</td></tr><tr><td>q3</td><td>-</td><td>q2</td></tr></tbody></table>		Next state		0	1	→ q0	q0,q1	q0	q1	q2	q1	q2	q3	q3	q3	-	q2	[L6][CO1]	[10M]
	Next state																				
	0	1																			
→ q0	q0,q1	q0																			
q1	q2	q1																			
q2	q3	q3																			
q3	-	q2																			

5		Explain the procedure adapted for minimization of finite automata. Minimize the following automata 	[L3][CO1]	[10M]
6	a	List out the identities of Regular expression.	[L1][CO1]	[5M]
	b	From the identities of RE, prove that i) $10+(1010)^*[\wedge+(1010)^*]=10+(1010)^*$ ii) $(1+100^*)+(1+100^*)(0+10^*)(0+10^*)^*=10^*(0+10^*)^*$	[L3][CO1]	[5M]
7		Explain the procedure adopted to convert Regular Grammar to Finite Automata and Convert the given Regular Grammar to Finite Automata $S \rightarrow aA/bB/a/b$ $A \rightarrow aS/bB/b$ $B \rightarrow aA/bS$	[L3][CO1]	[10M]
8	a	Construct an equivalent FA for the given regular expression using Top Down approach $10 + (0 + 11)0^* 1$	[L6][CO1]	[5M]
	b	Construct an equivalent FA for the given regular expression using Bottom up approach $(0+1)^*(00+11)(0+1)^*$	[L6][CO1]	[5M]
9	a	State the Pumping lemma and Prove that $L = \{a^i b^i \mid i \geq 0\}$ is not regular	[L3][CO1]	[6M]
	b	List out the various applications of Pumping lemma.	[L1][CO1]	[4M]
10	a	Discuss about the Closure properties of Regular Sets	[L2][CO1]	[5M]
	b	Find the regular expression for the following FA.	[L1][CO1]	[5M]
				

UNIT –IICONTEXT FREE GRAMMARS AND PUSHDOWN AUTOMATA

1	a	What is Context Free Grammar?	[L1][CO2]	[2M]
	b	State derivation tree with example.	[L1][CO2]	[2M]
	c	Define Ambiguous grammar.	[L1][CO2]	[2M]
	d	List out the methods of to simplify the CFG.	[L2][CO2]	[2M]
	e	Write the formal definition of Push Down Automata	[L2][CO2]	[2M]
2		Describe and Construct Leftmost derivation, Rightmost derivation and derivation tree for the string 0100110 using the given grammar $S \rightarrow 0S/1AA$ $A \rightarrow 0/1A/0B$ $B \rightarrow 1/0BB$	[L6][CO2]	[10M]
3	a	What is left recursion? Eliminate left recursion for the following grammar $E \rightarrow E+T/T$ $T \rightarrow T * F/F$ $F \rightarrow (E)/id$	[L3][CO2]	[5M]
	b	Show what you understand by Left factoring. Perform left factor for the grammar $A \rightarrow abB/aB/cdg/cdeB/cdfB$	[L3][CO2]	[5M]
4		Evaluate simplification of the following context free grammar. $S \rightarrow Aa/B$ $B \rightarrow a/bC$ $C \rightarrow a/\epsilon$	[L5][CO2]	[10M]
5	a	Remove the unit production from the grammar $S \rightarrow AB$ $A \rightarrow E$ $B \rightarrow C$ $C \rightarrow D$ $D \rightarrow b$ $E \rightarrow a$	[L3][CO2]	[5M]
	b	Remove ϵ productions from the grammar $S \rightarrow ABaC$ $A \rightarrow BC$ $B \rightarrow b/\epsilon$ $C \rightarrow D/\epsilon$ $D \rightarrow d$	[L3][CO2]	[5M]
6	a	Write the process adapted to convert the grammar into CNF?	[L2][CO2]	[4M]
	b	Convert the following grammar into CNF. $S \rightarrow bA/aB$ $A \rightarrow bAA/aS/a$ $B \rightarrow aBB/bS/a$	[L3][CO2]	[6M]
7		Define Greibach Normal Form. Convert the following grammar into Greibach Normal Form. $S \rightarrow AA/a$ $A \rightarrow SS/b$	[L3][CO2]	[10M]
8	a	Describe about acceptance of PDA.	[L2][CO2]	[4M]
	b	Construct a PDA which recognizes all strings that contain equal number of 0's and 1's.	[L6][CO2]	[6M]
9		Construct a DPDA to accept the language $L = \{WCWR / W \in (a,b)^+\}$ by empty stack and final state.	[L6][CO2]	[10M]
10		Explain the procedure to Construct an equivalent PDA from a CFG and adapt the same for the given grammar. $S \rightarrow aAB bBA$ $A \rightarrow bS a$ $B \rightarrow aS b$.	[L6][CO2]	[10M]
11		Evaluate the process adapted and convert the given PDA into an equivalent CFG. $\delta(q_0, a, z_0) \rightarrow (q_1, z_1 z_0)$ $\delta(q_0, b, z_0) \rightarrow (q_1, z_2 z_0)$ $\delta(q_1, a, z_1) \rightarrow (q_1, z_1 z_1)$ $\delta(q_1, b, z_1) \rightarrow (q_1, \lambda)$ $\delta(q_1, b, z_2) \rightarrow (q_1, z_2 z_2)$ $\delta(q_1, a, z_2) \rightarrow (q_1, \lambda)$ $\delta(q_1, \lambda, z_2) \rightarrow (q_1, \lambda)$ // accepted by the empty stack.	[L5][CO2]	[10M]

UNIT –III**TURING MACHINES AND INTRODUCTION TO COMPILERS**

1	a	State the formal definition of Turing Machine.	[L1][CO2]	[2M]
	b	Define Input buffering.	[L1][CO3]	[2M]
	c	What are the parts of Compilation process.	[L1][CO2]	[2M]
	d	Give the differences between compiler and interpreter.	[L1][CO2]	[2M]
	e	List out the properties of Recursive and Recursively enumerable languages.	[L1][CO2]	[2M]
2		Explain the various types of Turing machine with an example.	[L2][CO2]	[10M]
3	a	Describe Instantaneous Description(ID) of Turing Machine.	[L2][CO2]	[5M]
	b	Explain about the graphical notation of TM.	[L2][CO2]	[5M]
4	a	Illustrate the procedure adapted to convert RE to TM.	[L3][CO2]	[5M]
	b	Convert the given regular Expression $(a+b)^*(aa+bb)(a+b)^*$ to TM	[L3][CO2]	[5M]
5		Construct a Turing machine that recognizes the language $L=\{a^n b^n, n>1\}$. Show an ID for the string 'aabb' with tape symbols.	[L6][CO2]	[10M]
6		Analyse the phases of a compiler with neat diagram.	[L4][CO3]	[10M]
7		Design the compiler by using the source program position = initial + rate * 60.	[L6][CO3]	[10M]
8		Explain in detail about the role of lexical analyzer in Compiler Design.	[L2][CO4]	[10M]
9		Explain about input buffering in compilation process.	[L2][CO4]	[10M]

UNIT –IV
PARSERS AND INTERMEDIATE CODE GENERATION

1	a	List out various LR parsers in Compiler design.	[L1][CO5]	[2M]
	b	Write any two differences between Top-Down and Bottom -Up parser	[L1][CO5]	[2M]
	c	State LL(1) parser.	[L1][CO5]	[2M]
	d	Identify 3 techniques of Intermediate code generation.	[L1][CO5]	[2M]
	e	State the three address code.	[L1][CO5]	[2M]
2	a	Illustrate the rules to be followed in finding the FIRST and FOLLOW.	[L3][CO5]	[6M]
	b	Find FIRST and FOLLOW for the following grammar? $S \rightarrow A$ $A \rightarrow aB / Ad$ $B \rightarrow b$ $C \rightarrow g$	[L3][CO5]	[4M]
3	a	Construct RDP for the following grammar. $E \rightarrow i E'$ $E' \rightarrow + i E' \mid \varepsilon$	[L6][CO5]	[5M]
	b	Compare SLR , CLR and LALR.	[L6][CO5]	[5M]
4		Consider the grammar $E \rightarrow E+T/T, \quad T \rightarrow T * F / F, \quad F \rightarrow (E) \mid id$ Design predictive parsing table and check the given grammar is LL(1) or not?	[L6][CO5]	[10M]
5		Prepare Shift Reduce Parsing for the input string using the grammar $S \rightarrow (L) \mid a$ $L \rightarrow L, S \mid S$ a)(a,(a,a)) b)(a,a)	[L6][CO5]	[10M]
6		Construct the LR(0) items for the following Grammar $S \rightarrow L = R$ $S \rightarrow R$ $L \rightarrow * R$ $L \rightarrow id$ $R \rightarrow L$	[L6][CO5]	[10M]
7		Construct CLR Parsing table for the given grammar $S \rightarrow CC \quad C \rightarrow aC / d$	[L6][CO5]	[10M]
8		Design the LALR parser for the following Grammar $S \rightarrow AA \quad A \rightarrow aA \quad A \rightarrow b$	[L6][CO5]	[10M]
9		Analyse the Intermediate Code generation with an example.	[L4][CO5]	[10M]
10		Explain Representation of Three Address Codes and perform the same for the given expression: $(x + y) * (y + z) + (x + y + z)$	[L6][CO5]	[10M]

UNIT –V
CODE OPTIMIZATION AND CODE GENERATION

1	a	List out the optimization techniques of basic blocks	[L1][CO6]	[2M]
	b	Define Basic Block.	[L1][CO6]	[2M]
	c	Draw the DAG for the following statement. $a+b*c+d+b*c$	[L1][CO6]	[2M]
	d	Identify Loop optimization techniques.	[L1][CO6]	[2M]
	e	Write any three issues in the design of a code generator	[L1][CO6]	[2M]
2		Explain the peephole optimization Technique with examples.	[L2][CO6]	[10M]
3		Explain the following i) Basic blocks ii) Flow Graphs	[L2][CO6]	[10M]
4		Analyze different types of optimization techniques of basic blocks	[L4][CO6]	[10M]
5		Interpret the algorithm for the partition into Basic Blocks with an example.	[L4][CO6]	[10M]
6		Construct the DAG for the following basic blocks 1. $t1:=4*i$ 2. $t2:=a[t1]$ 3. $t3:=4*i$ 4. $t4:=b[t3]$ 5. $t5:=t2*t4$ 6. $t6:=prod+t5$ 7. $prod:=t6$ 8. $t7:=i+1$ 9. $i:=t7$ if $i \leq 20$ goto 1	[L6][CO6]	[10M]
7		Interpret the principles of source code optimization techniques to be considered during code generation.	[L3][CO6]	[10M]
8	a	Discuss about function preserving transformations(Local optimization).	[L2][CO6]	[5M]
	b	Describe about loop optimization technique.	[L2][CO6]	[5M]
9		Explain the issues to be handled when code generator is designed.	[L2][CO6]	[10M]
10	a	Analyze the different forms in target program.	[L4][CO6]	[5M]
	b	Analyze Simple code generator	[L4][CO6]	[5M]

Prepared by

Dr.R.M.Mallika, Dr.K.Hemabala