Course Code: 23CS0518 **R23**

SIDDHARTH INSTITUTE OF ENGINEERING & TECHNOLOGY:PUTTUR (AUTONOMOUS)

Siddharth Nagar, Narayanavanam Road – 517583

QUESTION BANK (DESCRIPTIVE)

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

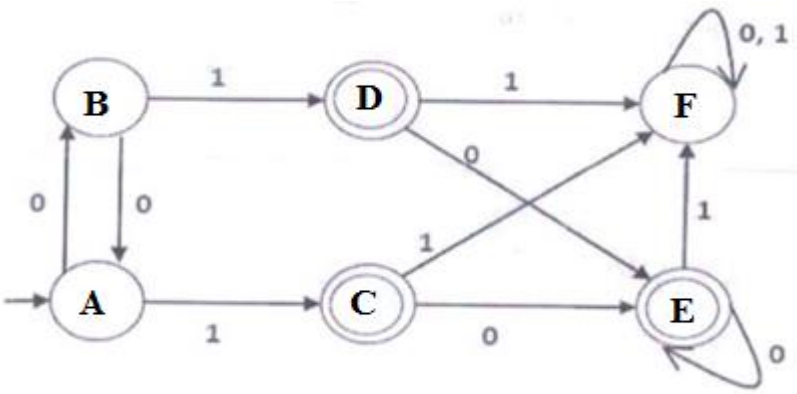
Course & Branch: III B.Tech – CSE **Year & Sem**: III &I Regulations: **R23**

UNIT –I
INTRODUCTION TO AUTOMATA AND REGULAR EXPRESSIONS

1	**a**	Define alphabet example.	**[L1][CO1]**	**[2M]**
	b	State what is Languages?	**[L1][CO1]**	**[2M]**
	c	Define Grammar.	**[L1][CO1]**	**[2M]**
	d	Define Finite Automata.	**[L1][CO1]**	**[2M]**
	e	Describe Regular Expression with example.	**[L1][CO1]**	**[2M]**
2		Define String. Describe String acceptance and check whether the given finite automata accept the given strings or not. (i) 0001 (ii) 1010	**[L2][CO1]**	**[5M]**
	b	Analyze and explain with example 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 0'sover {0, 1}.	**[L6][CO1]**	**[5M]**
4		Describe the procedure of conversion of NFA to DFA. Convert the given NFA to DFA.	**[L6][CO1]**	**[10M]**

States (Q)	Input Alphabtes 0	1
→q0	q1	q3
q1	q0	q2
(q2)	q3	q1
q3	q2	q0

	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)*[^+(1010)*]=10+(1010)* ii) (1+100*)+(1+100*)(0+10*)(0+10*)*=10*(0+10*)*	**[L3][CO1]**	**[5M]**
7		Explain the procedure adapted to convert Regular Grammar to Finite Automata and Convert the given Regular Grammar to Finite Automata S → aA/bB/a/b A→ aS/bB/b B→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 Pumping lemma for regular sets.	**[L1][CO1]**	**[4M]**
	b	Prove that $L = \{a^i b^i \mid i \geq 0\}$ is not regular	**[L3][CO1]**	**[6M]**
10	**a**	Discuss about the Closure properties of Regular Sets	**[L2][CO1]**	**[5M]**
	b	What are the applications of Pumping Lemma?	**[L1][CO1]**	**[5M]**

Course Code: 23CS0903 **R23**

UNIT –II
CONTEXT FREE GRAMMARS AND PUSHDOWN AUTOMATA

1	**a**	What is Context Free Language and Context Free Grammar?	**[L1][CO2]**	**[2M]**
	b	State what is derivation with example.	**[L1][CO2]**	**[2M]**
	c	Define Ambiguous grammar with examples.	**[L1][CO2]**	**[2M]**
	d	Describe Simplifying the Grammar.	**[L2][CO2]**	**[2M]**
	e	State 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→0S/1AA A→0/1A/0B B→1/0BB	**[L6][CO2]**	**[10M]**
3	**a**	What is left recursion? Eliminate left recursion for the following grammar E→E+T/T T→T*F/F F→(E)/id	**[L3][CO2]**	**[5M]**
	b	Show what you understand by Left factoring. Perform left factor for the grammar A→abB/aB/cdg/cdeB/cdfB	**[L3][CO2]**	**[5M]**
4		Evaluate simplification of the following context free grammar. S→ Aa /B B→ a/bC C→a / ϵ	**[L5][CO2]**	**[10M]**
5	**a**	Remove the unit production from the grammar S→AB A→E B→C C→D D→b E→a	**[L3][CO2]**	**[5M]**
	b	Remove ϵ productons from the grammar S→ABaC A→BC B→b/ ϵ C→D/ϵ D→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→ bA/aB A→bAA/aS/a B→aBB/bS/a	**[L3][CO2]**	**[6M]**
7		Define Greibach Normal Form. Convert the following grammar into Greibach Normal Form. S→AA/a A→SS/b	**[L3][CO2]**	**[10M]**
	a	Describe about acceptance of PDA.	**[L2][CO2]**	**[4M]**
8	**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 ∈ (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→aAB | bBA A→bS | a B→aS | b.	**[L6][CO2]**	**[10M]**
11		Evaluate the process adapted and convert the given PDA into an equivalent CFG. δ (q0,a0,z0)→(q1,z1z0) δ(q0,b,z0)→(q1,z2z0) δ(q1,a,z1)→(q1, z1z1) δ(q1,b,z1)→(q1, λ) δ(q1,b,z2)→(q1,z2z2) δ(q1,a,z2)→(q1, λ) δ(q1, λ,z2)→(q1, λ) // accepted by the empty stack.	**[L5][CO2]**	**[10M]**

Course Code: 23CS0903 **R23**

UNIT –III
TURING MACHINES AND INTRODUCTION TO COMPILERS

1	**a**	State Turing machine.	**[L1][CO2]**	**[2M]**
	b	Define Compiler	**[L1][CO3]**	**[2M]**
	c	List all the phases of compiler	**[L1][CO2]**	**[2M]**
	d	Give the differences between compiler and interpreter.	**[L1][CO2]**	**[2M]**
	e	List the different types of Turing Machine.	**[L1][CO2]**	**[2M]**
2		Explain the various types of Turing machine.	**[L2][CO2]**	**[10M]**
3	**a**	Describe Instantaneous Description 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		Describe in detail the phases of a compiler with neat diagram.	**[L2][CO3]**	**[10M]**
7		Design the compiler by using the source program position = intial + rate * 60.	**[L6][CO3]**	**[10M]**
8		Explain in detail about the role of lexical analyzer in Compiler Design.	**[L2][CO4]**	**[10M]**
9		What is input buffering? Explain its purpose and how it works	**[L2][CO4]**	**[10M]**

Course Code: 23CS0903 **R23**

UNIT –IV
PARSERS AND INTERMEDIATE CODE GENERATION

1	**a**	What is meant by Non-recursive predictive parsing?	**[L2][CO5]**	**[2M]**
	b	Analyze the difference between Top-Down and Bottom -Up parser	**[L1][CO5]**	**[2M]**
	c	List the types of parsers available in compilers	**[L1][CO5]**	**[2M]**
	d	Define augmented grammar.	**[L1][CO5]**	**[2M]**
	e	Describe FIRST and FOLLOW with example	**[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? E→ E+T/T T→T*F/F F→(E)/id	**[L3][CO5]**	**[4M]**
3		Consider the grammar S→AB | ABad A→d E →b D→b | ε B→c Construct the predictive parse table and check whether the given grammar is LL(1) or not.	**[L6][CO5]**	**[10M]**
4		Consider the grammar E→E+T/T, T→T*F/F, F→(E)|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→(L)|a L→L,S|S a)(a,(a,a)) b)(a,a)	**[L6][CO5]**	**[10M]**
6		Construct the LR(0) items for the following Grammar S→L=R S→R L→*R L→id R→L	**[L6][CO5]**	**[10M]**
7		Construct CLR Parsing table for the given grammar S→CC C→aC/d	**[L6][CO5]**	**[10M]**
8		Design the LALR parser for the following Grammar S → AA A → aA A →b	**[L6][CO5]**	**[10M]**
9		Analyse different types of Intermediate Code 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]**

Course Code: 23CS0903 **R23**

UNIT –V
CODE OPTIMIZATION AND CODE GENERATION

1	**a**	List the optimization techniques of basic blocks	**[L1][CO6]**	**[2M]**
	b	Define DAG with example	**[L1][CO6]**	**[2M]**
	c	Create the DAG for following statement. a+b*c+d+b*c	**[L6][CO6]**	**[2M]**
	d	Discuss about machine dependent optimization	**[L1][CO6]**	**[2M]**
	e	List all the 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	**[L3][CO6]**	**[10M]**
4		Analyse different types of optimization techniques of basic blocks	**[L4][CO6]**	**[10M]**
5		List out the properties of global data flow analysis and explain it.	**[L2][CO6]**	**[5M]**
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<=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.	**[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]**	**[5M]**
10	**a**	Analyse 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, K.Hema