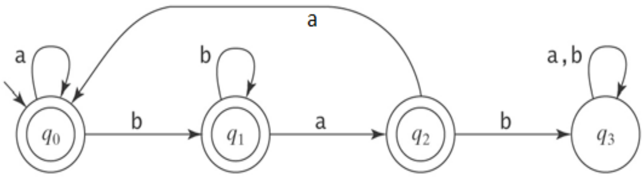
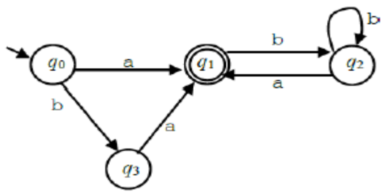
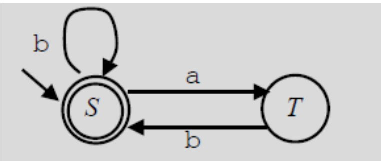
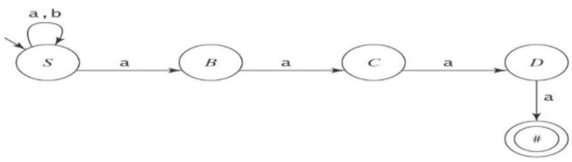
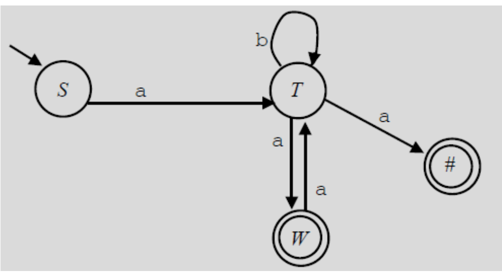
CMR INSTITUTE OFTECHNOLOGY USN

Second Internal Test

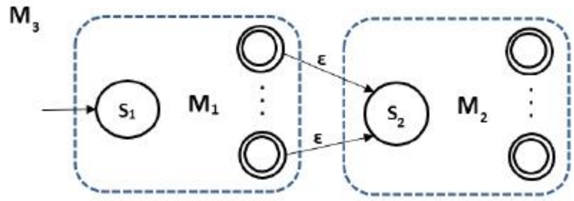
Sub:	Automata Theory and Computability							Code:	15CS54
Date:	16 / 10 / 2018	Duration:	90 mins	Max Marks:	50	Sem:	V	Branch:	ISE

Answer **ANY 5 Full** Questions

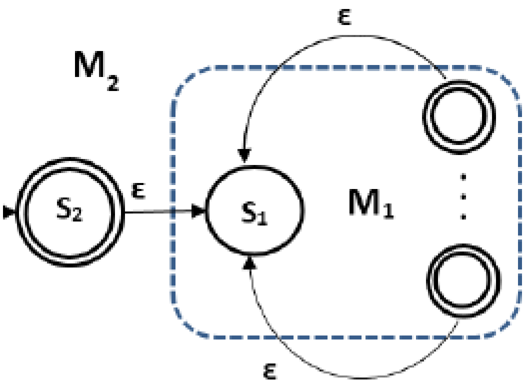
		Marks	OBE CO	OBE RBT
1 (a)	Write a regular expression to describe each of the following languages: i) {*w* ∈ {0-9}* : *w* corresponds to the decimal encoding, without leading 0's, of an odd natural number}. (ε ∪ ((1-9)(0-9)*))(1 ∪ 3 ∪ 5 ∪ 7 ∪ 9) ii) {*w* ∈ {a, b}* : *w* has both aa and bb as substrings}. (a ∪ b)* aa (a ∪ b)* bb (a ∪ b)* ∪ (a ∪ b)* bb	[04] 2 Marks each	CO2	L3
(b)	Convert following FSM to RE.  (a ∪ bb*aa)* (ε ∪ bb*(a ∪ ε)).	[03] Partially Correct Give 2 marks	CO2	L3
(c)	Indicate, for each of the following regular expressions, whether it correctly describes L:  a. (a ∪ ba)bb*a. b. (ε ∪ b)a(bb*a)*. c. ba∪ ab*a. d. (a ∪ ba)(bb*a)*. a) no; b) yes; c) no; d) yes.	[03] At least Three correct give 3 marks	CO3	L3
2 (a)	Briefly explain the applications of regular expression. Email, IP addressing Legal Passwords XML	[02] 2 with examples explanation can be given.	CO1	L2
(b)	Define regular expression. Write the regular expression for the following language. (i) L={$a^n b^m$|n<=4, m>=2} (ε+a+aa+aaa+aaaa)bbb* (ii) Strings of 0's and 1's having at least two 0's (0+1)*0 (0+1)*0 (0+1)*	[04] 2 marks each	CO3	L3
(c)	Simplify the following Regular expression. (i)a ((a ∪ b)(b ∪ a))* ∪ a ((a ∪ b) a)* ∪ a ((b ∪ a) b)*. a ((a ∪ b)(b ∪ a))*. (ii) (a ∪ b)*a* ∪ b. (a ∪ b)*.	[04] 2 marks each	CO3	L3
3 (a)	Show the regular language for the following Language.	[04] Partial correct	CO3	L3

	$\{w \in \{a, b\}^* : w$ does not end in aa$\}$. $\varepsilon \cup a \cup (a \cup b)^* (ba \cup ab \cup bb)$	expression can be given 2-3 marks		
(b)	Let $L = \{w \in \{a, b\}^* :$ every a in w is immediately followed by at least one b$\}$. (i) Write a regular expression that describes L. $(ab \cup b)^*$ (ii) Write a regular grammar that generates L. $S \rightarrow bS$ $S \rightarrow aT$ $S \rightarrow \varepsilon$ $T \rightarrow bS$ (iii) Construct an FSM that accepts L. 	[06] 2+2+2 marks	CO3	L3
4(a)	Give the regular grammar for the FSM in figure.  $L = \{w \in \{a, b\}^* : w$ ends with the pattern aaaa$\}$. $S \rightarrow aS$ $S \rightarrow bS$ $S \rightarrow aB$ $B \rightarrow aC$ $C \rightarrow aD$ $D \rightarrow a$	[05] Each grammar rule 1 mark can be given	CO3	L3
(b)	Construct FSM for the following regular grammar. $S \rightarrow aT \quad T \rightarrow bT \quad T \rightarrow a \quad T \rightarrow aW \quad W \rightarrow \varepsilon \quad W \rightarrow aT$ 	[05] States correct - 2 marks Transition correct-3 marks	CO4	L4
5(a)	If L_1 and L_2 are regular languages prove that $L_1 \cup L_2$, $L_1.L_2$, and L_1^* are also regular languages.	[05] Union =2 marks Concatenation =2 marks Kleene Star 1 mark	CO4	L3

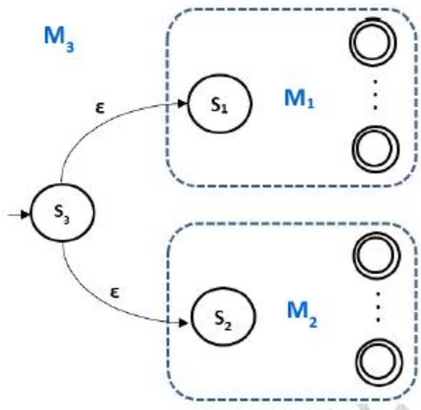
If α is the regular expression $\beta\gamma$ and if both $L(\beta)$ and $L(\gamma)$ are regular, then we construct $M_3 = (K_3, \Sigma, \delta_3, s_3, A_3)$ such that $L(M_3) = L(\alpha) = L(\beta)L(\gamma)$. If necessary, rename the states of M_1 and M_2 so that $K_1 \cap K_2 = \varnothing$. We will build M_3 by connecting every accepting state of M_1 to the start state of M_2 via an ε-transition. M_3 will start in the start state of M_1 and will accept iff M_2 does. So $M_3 = (K_1 \cup K_2, \Sigma, \delta_3, s_1, A_2)$, where $\delta_3 = \delta_1 \cup \delta_2 \cup \{((q, \varepsilon), s_2) : q \in A_1\}$.

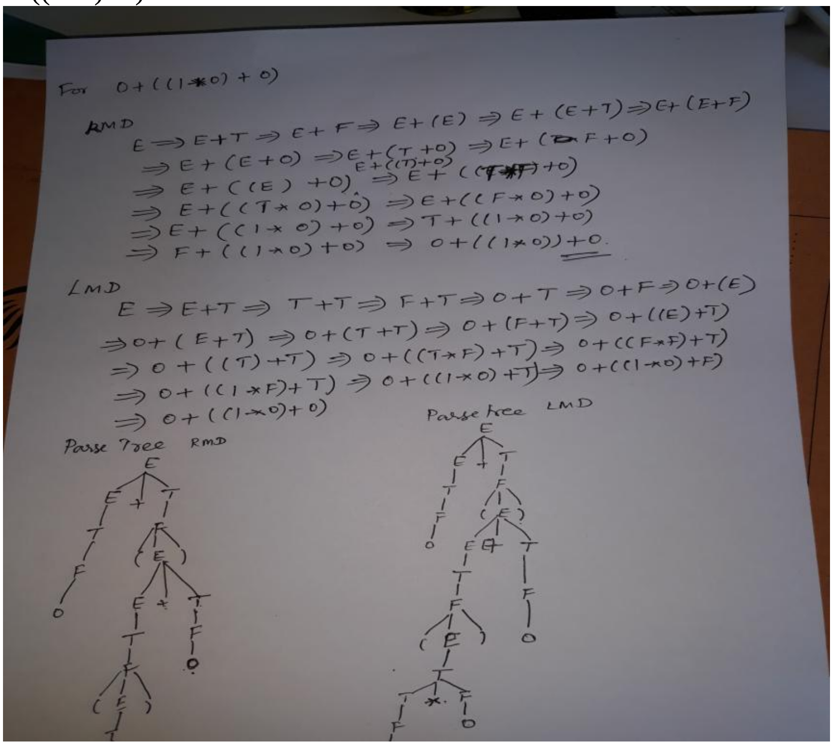


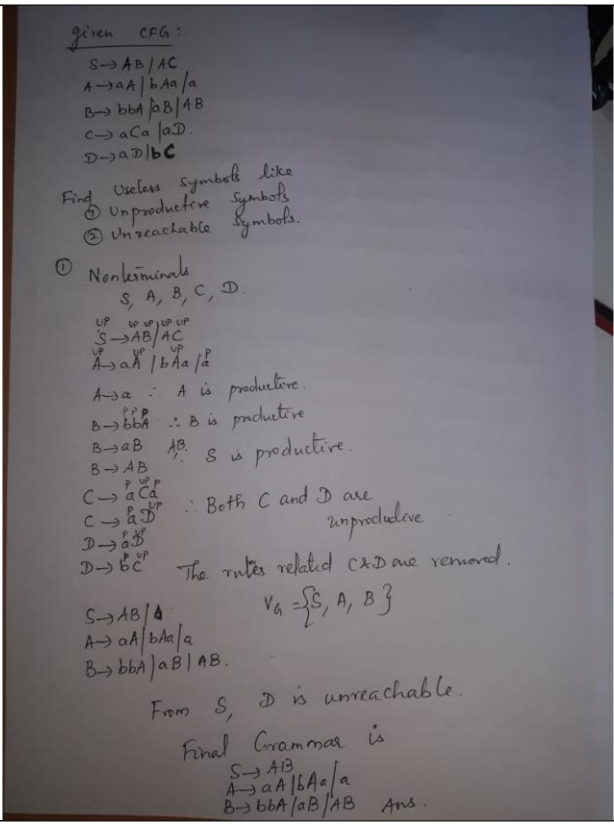
If α is the regular expression β^* and if $L(\beta)$ is regular, then we construct $M_2 = (K_2, \Sigma, \delta_2, s_2, A_2)$ such that $L(M_2) = L(\alpha) = L(\beta)^*$. We will create a new start state s_2 and make it accepting, thus assuring that M_2 accepts ε. (We need a new start state because it is possible that s_1, the start state of M_1, is not an accepting state. If it isn't and if it is reachable via any input string other than ε, then simply making it an accepting state would cause M_2 to accept strings that are not in $(L(M_1))^*$.) We link the new s_2 to s_1 via an ε-transitions. Finally, we create ε-transitions from each of M_1's accepting states back to s_1. So $M_2 = (\{s_2\} \cup K_1, \Sigma, \delta_2, s_2, \{s_2\} \cup A_1)$, where $\delta_2 = \delta_1 \cup \{((s_2, \varepsilon), s_1)\} \cup \{((q, \varepsilon), s_1) : q \in A_1\}$.

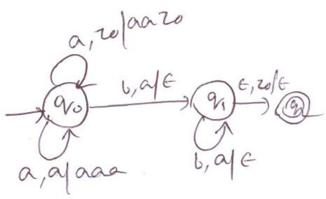


- If α is the regular expression $\beta \cup \gamma$ and if both $L(\beta)$ and $L(\gamma)$ are regular, then we construct $M_3 = (K_3, \Sigma, \delta_3, s_3, A_3)$ such that $L(M_3) = L(\alpha) = L(\beta) \cup L(\gamma)$. If necessary, rename the states of M_1 and M_2 so that $K_1 \cap K_2 = \varnothing$. Create a new start state, s_3, and connect it to the start states of M_1 and M_2 via ε-transitions. M_3 accepts iff either M_1 or M_2 accepts. So $M_3 = (\{s_3\} \cup K_1 \cup K_2, \Sigma, \delta_3, s_3, A_1 \cup A_2)$, where $\delta_3 = \delta_1 \cup \delta_2 \cup \{((s_3, \varepsilon), s_1), ((s_3, \varepsilon), s_2)\}$.



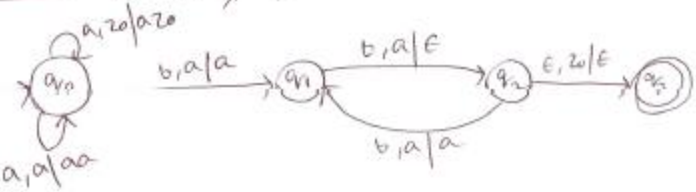
(b)	Consider the CFG with productions E → E+T | T T → T*F |F F → (E) | 0 | 1 Write Left Most Derivation, Right Most Derivation and Parse tree for the string 0+((1*0)+0)  	[05] RMD-1.5 marks LMD-1.5 marks Parse Tree 2*1 = 2 marks	CO3	L3
6.	Define a Context-free grammar (CFG). Write the CFG for the following languages. (a) L={$a^{2n}b^n$|n≥1} (b) L={$a^ib^jc^k$| j=i+k and i,k≥1} (c) L={w c w^R|w Є {a,b}*} (a) $L = \{a^{2n} b^n \mid n \geq 1\}$ (2M) S → aaSb | aab (b) $L = \{a^i b^j c^k \mid j = i+k \text{ and } i,k \geq 1\}$ (3M) $a^i\ b^j\ c^k$ $a^i\ b^{i+k}\ c^k$ $\underbrace{a^i\ b^i}\ \underbrace{b^k\ c^k}$ S → AB A → aAb | ab B → bBc | bc (c) $L = \{wcw^R \mid w \in \{a,b\}^*\}$ (3M) S → aSa | bSb | c	[10] a) 3 marks b) 4 marks c) 3 marks	CO2	L4
7(a)	Simplify the following CFG S→ AB |AC A→aA |bAa|a, B→bbA|aB|AB C→aCa|aD D→aD|bC	[05] Removing Unproductive Symbols 3 marks Removing	CO4	L3

	 given CFG: S → AB / AC A → aA | bAa / a B → bbA / aB | AB C → aCa / aD. D → aD | bC Find Useless symbols like ① Unproductive symbols ② Unreachable symbols. ① Nonterminals S, A, B, C, D. S → AB / AC A → aA / bAa / a A → a ∴ A is productive. B → bbA ∴ B is productive B → aB AB ∴ S is productive. B → AB C → aCa C → aD ∴ Both C and D are unproductive D → aD D → bC The rules related C & D are removed. S → AB / A A → aA / bAa / a B → bbA / aB | AB. $V_G = \{S, A, B\}$ From S, D is unreachable. Final Grammar is S → AB A → aA / bAa / a B → bbA / aB / AB Ans.	unreachable symbols: 2 marks		
(b)	For the following grammar G, show that G is ambiguous. Then find an equivalent grammar that is not ambiguous. a) $(\{S, A, B, T, \mathtt{a}, \mathtt{c}\}, \{\mathtt{a}, \mathtt{c}\}, R, S)$, where $R = \{S \to AB, S \to BA, A \to \mathtt{a}A, A \to \mathtt{ac},$ $B \to T\mathtt{c}, T \to \mathtt{a}T, T \to \mathtt{a}\}$. Both A and B generate $\mathbf{a^+c}$. So any string in L can be generated two ways. The first begins $S \Rightarrow AB$. The second begins $S \Rightarrow BA$. The easy fix is to eliminate one of A or B. We pick B to eliminate because it uses the more complicated path, through T. So we get: $G' = \{\{S, A, \mathtt{a}, \mathtt{c}\}, \{\mathtt{a}, \mathtt{c}\}, R, S\}$, where $R = \{S \to AA, A \to \mathtt{a}A, A \to \mathtt{ac}\}$. G' is unambiguous. Any derivation in G' of the string $\mathtt{a}^n\mathtt{c}$ must be of the form: $S \Rightarrow AA \Rightarrow^{n-1} \mathtt{a}^{n-1}A \Rightarrow \mathtt{a}^{n-1}\mathtt{ac}$. So there is only one leftmost derivation in G' of any string in L.	[05]	CO4	L4
8(a)	Convert the following grammar into Chomsky Normal Form. $S \to ABC$ $A \to \mathtt{a}C \mid D$ $B \to \mathtt{b}B \mid \varepsilon \mid A$ $C \to A\mathtt{c} \mid \varepsilon \mid C\mathtt{c}$ $D \to \mathtt{aa}$ **Answer: 4 Steps:** **Remove** ε rules, Remove Unit Production, Remove Mixed Production and Remove Long Production	[05] Writing 4 steps : 1 mark Each step carries 1 mark	CO3	L3

	$S \to ABC$ $A \to aC \mid D$ $B \to bB \mid \varepsilon \mid A.$ $C \to Ac \mid \varepsilon \mid Cc$ $D \to aa.$ ① Remove ε rules. $N = \{B, C\}$ $G^{*} = S \to ABC$ $A \to aC \mid D,\ B \to bB \mid A \mid \varepsilon,\ C \to Ac \mid Cc \mid \varepsilon,\ D \to aa$ G' $S \to ABC \mid AB \mid AC \mid A$ $A \to aC \mid a \mid D$ $B \to bB \mid b \mid A$ $C \to Ac \mid C$ $D \to aa$ After remove ε rules and doing modification of rules. ② Unit Production removal Remove $S \to A$ add $S \to aC \mid a \mid D$. Remove $A \to D$ add $A \to aa$ Remove $B \to A$ add ~~$B \to Ac \mid c$~~ $B \to aC \mid a \mid D$ Remove $S \to D$ Add $S \to aa$ Remove $B \to D$ Add $B \to aa.$ Now G'' $S \to ABC \mid AB \mid AC \mid aC \mid a \mid aa$ $A \to aC \mid aa \mid a$ $B \to bB \mid aC \mid a \mid b \mid aa$ $C \to Ac \mid C$ $D \to aa$ ③ Remove mixed production $S \to ABC \mid AB \mid AC \mid T_aC \mid T_a \mid T_aT_a$ $A \to T_aC \mid T_aT_a \mid T_a$ $B \to T_bB \mid T_aC \mid T_aT_a \mid T_a \mid T_b$ $C \to AT_c \mid T_c$ $D \to T_aT_b$ $T_a \to a$ $T_b \to b$ $T_c \to c$ ④ Remove longer Production $S \to AS_1 \mid AB \mid AC \mid T_aC \mid T_a \mid T_aT_a$ $S_1 \to BC$ $A \to T_aC \mid T_aT_a \mid T_a$ $B \to T_bB \mid T_aC \mid T_aT_a \mid T_a \mid T_b$ $C \to AT_c \mid T_c$ $D \to T_aT_b$ $T_a \to a$ $T_b \to b$ $T_c \to c$. Ans.			
(b)	Design a PDA for the language $L=\{a^nb^{2n} \mid n\geq 1\}$. Show the ID for the input string w=aabbbb method 1 $\delta(q_0, a, z_0) = (q_0, aaz_0)$ $\delta(q_0, a, a) = (q_0, aaa)$ $\delta(q_0, b, a) = (q_1, \epsilon)$ $\delta(q_1, b, a) = (q_1, \epsilon)$ $\delta(q_1, \epsilon, z_0) = (q_2, \epsilon)$  	[05] PDA 3 marks ID= 2 marks	CO4	L3

<table>
<tr><td></td><td>Method 2

aabbbb



$\delta(q_0, a, Z_0) = (q_0, aZ_0)$
$\delta(q_0, a, a) = (q_0, aa)$
$\delta(q_0, b, a) = (q_1, a)$
$\delta(q_1, b, a) = (q_2, \epsilon)$
$\delta(q_2, b, a) = (q_1, a)$
$\delta(q_2, \epsilon, Z_0) = (q_3, \epsilon)$

ID for $w = aabbbb$ using method 1
$(q_0, aabbbb, Z_0) \vdash (q_0, abbbb, aaZ_0)$
$\vdash (q_0, bbbb, aaaaZ_0)$
$\vdash (q_1, bbb, aaaZ_0) \vdash (q_1, bb, aaZ_0)$
$\vdash (q_1, b, aZ_0) \vdash (q_1, \epsilon, Z_0)$
$\vdash (q_2, \epsilon, \epsilon)$ Accepted</td><td></td><td></td><td></td></tr>
<tr><td>9(a)</td><td>Prove that the following grammar is unambiguous: $L=\{a^nb^n \mid n\geq 0\}$ and
$G = \{\{S, a, b\}, \{a, b\}, R, S\}$, where:
$R = \{S \rightarrow aSb$
$S \rightarrow \varepsilon\}$.

We now show that G is correct. We first show that every string w in $L(G)$ is in A^nB^n: Let st be the working string at any point in a derivation in G. We need to define I so that it captures the two features of every string in A^nB^n: The number of a's equals the number of b's and the letters are in the correct order. So we let I be:

$$(\#_a(st) = \#_b(st)) \wedge (st \in a^*(S \cup \varepsilon)b^*).$$
Now we prove:
• I is true when $st = S$: In this case, $\#_a(st) = \#_b(st)) = 0$ and st is of the correct form.
• If I is true before a rule fires, then it is true after the rule fires: To prove this, we consider the rules one at a time and show that each of them preserves I. Rule (1) adds one a and one b to st, so it does not change the difference between the number of a's and the number of b's. Further, it adds the a to the left of S and the b to the right of S, so if the form constraint was satisfied before applying the rule it still is afterwards. Rule (2) adds nothing so it does not change either the number of a's or b's or their locations.
• If I is true and st contains only terminal symbols, then $st \in A^nB^n$: In this case, st possesses the three properties required of all strings in A^nB^n: They are composed only of a's and b's, $(\#_a(st) = \#_b(st))$, and all a's come before all b's.</td><td>[07]

Loop Invariant: 2 marks
3 Proving Point 3 marks
Proof by Induction
2 marks</td><td>CO4</td><td>L3</td></tr>
</table>

	Next we show that every string w in A^nB^n can be generated by G: Every string in A^nB^n is of even length, so we will prove the claim only for strings of even length. The proof is by induction on $\|w\|$: • Base case: If $\|w\| = 0$, then $w = \varepsilon$, which can be generated by applying rule (2) to S. • Prove: If every string in A^nB^n of length k, where k is even, can be generated by G, then every string in A^nB^n of length $k + 2$ can also be generated. Notice that, for any even k, there is exactly one string in A^nB^n of length k : $a^{k/2}b^{k/2}$. There is also only one string of length $k + 2$, namely $aa^{k/2}b^{k/2}b$, that can be generated by first applying rule (1) to produce aSb, and then applying to S whatever rule sequence generated $a^{k/2}b^{k/2}$. By the induction hypothesis, such a sequence must exist.			
(b)	What is meant by rejecting computation in PDA? Give example for it. A computation C of M is a ***rejecting computation*** iff: • $C = (s, w, \varepsilon) \vdash_M^* (q, w', \alpha)$, • C is not an accepting computation, and • M has no moves that it can make from (q, ε, α). M ***rejects*** a string w iff all of its computations reject.	[03] 2 marks for definition. One example 1 mark	CO3	L2