Branch: IT	Course: Automata Theory
Year/ Semester: SE/III	Course code: ITC304
Time: 03 hours	Marks: 80

Solution

Q.1 (30 Marks)

A. Describe Moore machine with all tuples in detail (5 Marks)

- Definition – 1 mark
- Five-tuple – 1 mark
- Explanation of tuples – 2 marks
- Output depends on state – 1 mark

B. Explain Right Linear and Left Linear Grammars (5 Marks)

- Definition of right-linear grammar – 1 mark
- Production form – 1 mark
- Definition of left-linear grammar – 1 mark
- Production form – 1 mark
- Language generated – 1 mark

C. Design a Moore machine to count occurrences of substring `bba` (5 Marks)

- State definitions – 1 mark
- Transition table/diagram – 2 marks
- Output assignment – 1 mark
- Overlapping occurrences – 1 mark

D. Regular expression for strings not ending in `01` (5 Marks)

- Identification of cases – 2 marks
- Regular expression – 2 marks
- Correctness – 1 mark

E. Applications of FA, PDA and TM (5 Marks)

- FA applications – 1.5 marks
- PDA applications – 1.5 marks
- TM applications – 1.5 marks
- Examples – 0.5 mark

F. Steps for converting CFG to CNF (5 Marks)

- New start symbol – 1 mark
- Elimination of ε-productions – 1 mark
- Elimination of unit productions – 1 mark
- Removal of useless symbols – 1 mark
- Conversion to CNF production forms – 1 mark

Q.2 (10 Marks)

A. Design DFA for $(0+1)^*010(0+1)^*$

- State definitions – 2 marks
- Transition table – 3 marks
- State diagram – 3 marks
- Initial and final states – 1 mark
- Correctness – 1 mark

B. Design NFA for strings containing `aaa` or `aba` and convert to DFA

- NFA diagram – 3 marks
- NFA transitions – 2 marks
- Subset construction – 2 marks
- Equivalent DFA – 2 marks
- Final states – 1 mark

Q.3 (10 Marks)

A. Convert the given CFG into GNF

- Substitution of productions – 2 marks
- Elimination of left recursion, if required – 3 marks
- Terminal-leading productions – 3 marks
- Final GNF – 2 marks

B. Show that the given grammar is ambiguous

- Definition of ambiguity – 2 marks
- Selection of string – 1 mark
- First derivation/parse tree – 3 marks
- Second derivation/parse tree – 3 marks
- Conclusion – 1 mark

Q.4 (10 Marks)

A. Construct PDA for $L = \{a^n b^m c^n \mid m,n >= 0\}$

State definitions – 2 marks

- Push operation – 2 marks
- Processing `b` symbols – 2 marks
- Pop operation – 2 marks
- Acceptance condition – 2 marks

B. Design PDA for $L = \{ 0^n 1 2^n \mid n >= 1\}$

- State definitions – 2 marks
- Push operation – 2 marks
- Processing separator `1` – 2 marks
- Pop operation – 2 marks
- Acceptance condition – 2 marks

Q.5 (10 Marks)

A. Design TM for $\{a^n b^n C^n \mid n >= 1\}$ Can DFA be designed?

- State and tape symbol definitions – 2 marks
- Marking `a` – 2 marks
- Matching and marking `b` – 2 marks
- Matching and marking `c` – 2 marks
- Acceptance/rejection – 1 mark
- DFA justification – 1 mark

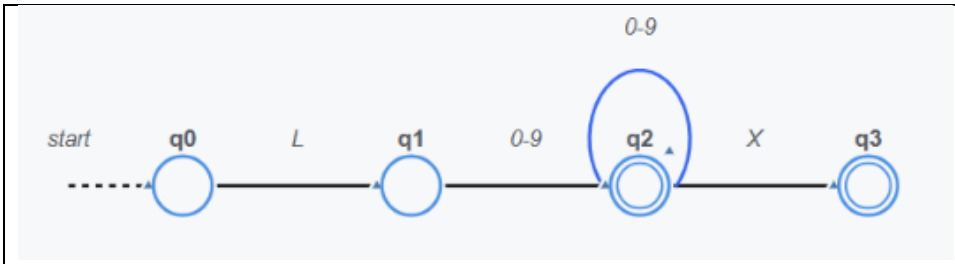
B. Design TM for palindromes over {0,1} and describe all tuples

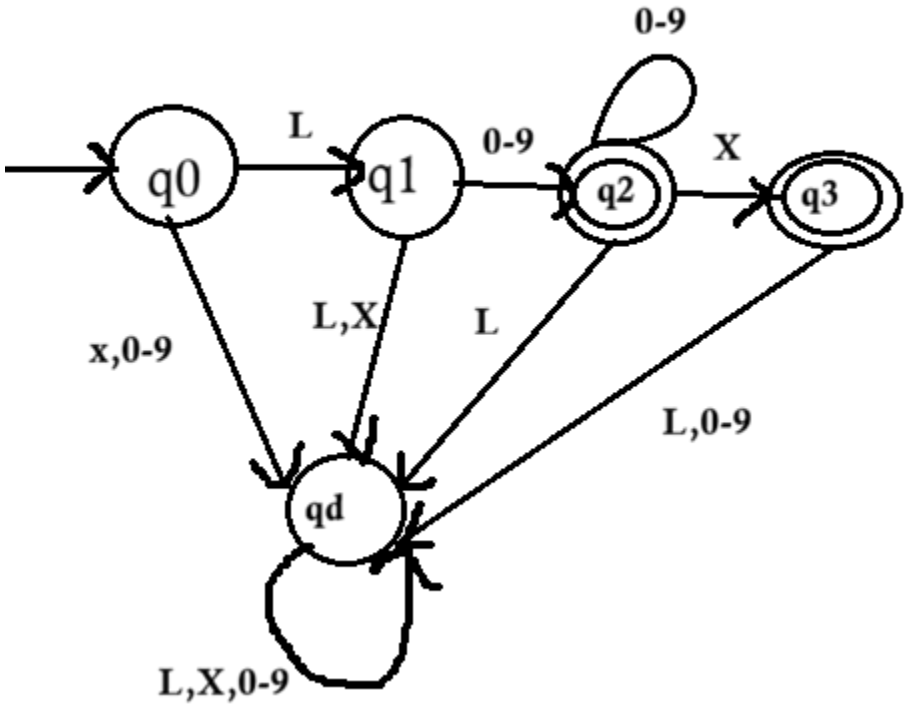
- Seven-tuple definition – 2 marks
- Mark leftmost symbol – 2 marks
- Compare rightmost symbol – 2 marks
- Repeat matching process – 2 marks
- Acceptance/rejection – 2 marks

Q.6 (10 Marks)

A.

RE: $L[0\text{-}9]^+(X+€)$



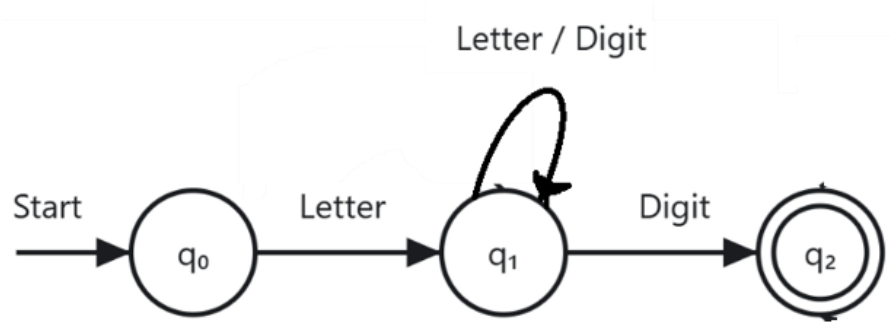


Transition table for the same

String	Processing Trace	Status
L52	• Start at q_0 • Read L → moves to q_1 • Read 5 → moves to q_2 • Read 2 → loops back to q_2	Accepted
LX	• Start at q_0 • Read L → moves to q_1 • Next character is X, but q_1 only accepts a digit $(0-9)$.	Rejected
L52X	• Start at q_0 • Read L → moves to q_1 • Read 5 → moves to q_2 • Read 2 → loops back to q_2 • Read X → moves to q_3	Accepted

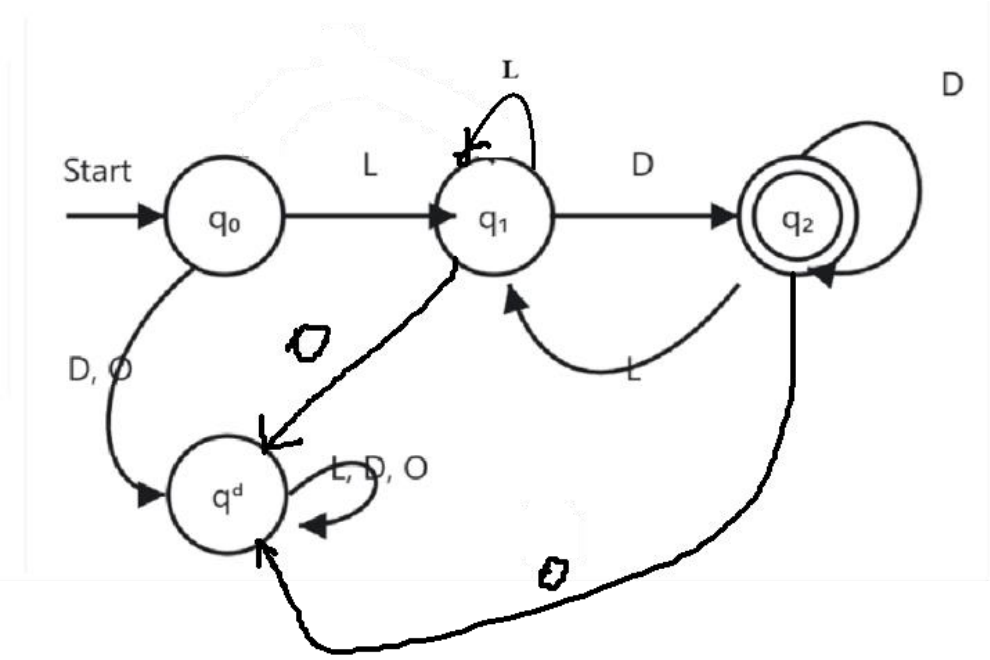
OR

B. RE: [a-zA-Z][a-zA-Z0-9]*[0-9]



- L: Any letter (`a-z`, `A-Z`)
- D: Any digit (`0-9`)
- O: Any other character

State	Letter	Digit
→q0	{q1}	∅
q1	{q1}	{q2}
*q2	{q1}	{q2}



Present State	Letter	Digit	Other
→ q0	q1	Dead	Dead
q1	q1	q2	Dead
*q2	q1	q2	Dead
Dead	Dead	Dead	Dead

Input String	NFA State Path Trace	Outcome
A7	$q_0 \xrightarrow{A} q_1 \xrightarrow{7} q_2$	Accepted (Valid)
7A	$q_0 \xrightarrow{7} \emptyset$ (Stuck)	Rejected (Invalid)
abC2	$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_1 \xrightarrow{C} q_1 \xrightarrow{2} q_2$	Accepted (Valid)