B.Tech.

Fourth Semester Examination 2009-10

Theory of Automata and Formal Language

Note : (i) This question paper contains Five questions.
(ii) Attempt all questions.

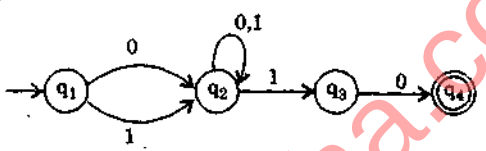
Q. 1. Attempt any four of the following :

(4 × 5 = 20)

Q. 1. (a) Draw a DFA that accepts set of all strings of 0 and 1 that end in the set two same symbols.

Ans. The set of all string ending in 00

$= (0 + 1)^* 00$



State/Σ	0	1
$\rightarrow q_1$	q_2	q_1
q_2	q_1, q_3	q_2
q_3	q_4	—
(q_4)	—	—

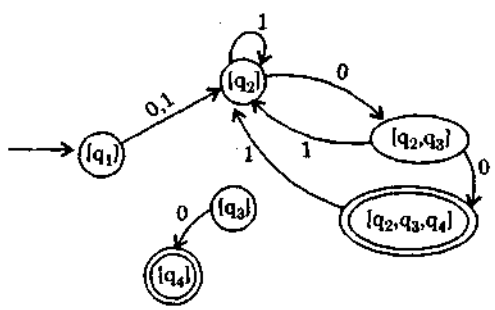
Now we can construct DFA for a above NFA

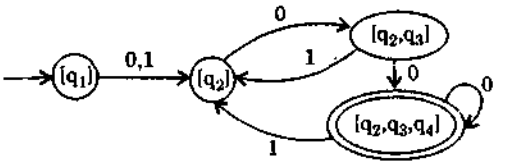
Transition table

State/Σ	0	1
$[q_1]$	$[q_2]$	$[q_2]$
$[q_2]$	$[q_2, q_3]$	$[q_2]$
$[q_3]$	$[q_4]$	—
$[q_2, q_3]$	$[q_2, q_3, q_4]$	$[q_2]$
$[q_2, q_3, q_4]$	$[q_2, q_3, q_4]$	$[q_2]$
$[q_4]$	—	—

New transition diagram for DFA

where $Q' = [q_1], [q_2], [q_3], [q_2, q_3], [q_4],$ $[q_2, q_3, q_4]$





This is the final DFA which accepts $(0 + 1)^*00$.

Q. 1. (b) Write down procedure for converting an NFA to its equivalent of A.

Ans. Equivalent DFA of the given NFA as

		1
$\rightarrow p$	$\{q, s\}$	$\{q\}$
$\{q, s\}$	$\{q, s, r\}$	$\{q, r\}$
$\{q, r, s\}$	$\{q, s, r\}$	$\{p\}$
$\{q, r, s\}$	$\{q, r, s\}$	$\{p\}$

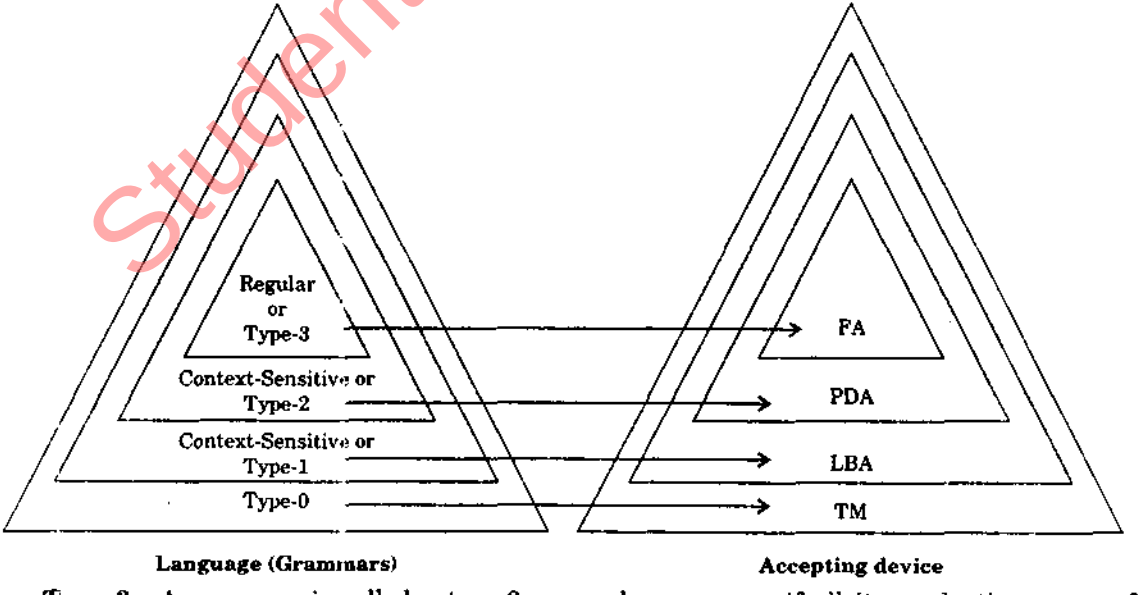
Q. 1. (c) Give the Chomsky Hierarchy of grammars specifically giving form of production rules in each class of grammar.

Ans. Chomsky classified the grammars into four types in terms of production (types 0-3).

Type-0 : A type-0 grammar in any phrase structure grammar without any restriction. In a production of the form $\phi A \psi \rightarrow \phi \propto \psi$ where A is a variable, ϕ is called the left context ψ the right context, and $\phi \propto \psi$ the replacement string.

Type-1 : A grammar is called type-1 or context sensitive context department if all its production are type-1 production. A production if $a \neq \wedge$. In type-1 productions, erasing of A is not permitted.

Type-2 : A type-2 production in a production of the form $A \rightarrow \alpha$, where $A \in V_N$ and $\alpha \in (V_N \cup \Sigma)^*$. In other words, the LHS has no left context or right context.



Type-3 : A grammar is called a type-3 or regular grammar if all its production or type-3 production. A production $S \rightarrow \wedge$ is allowed in type-3 grammar, but in care S does not on the right hand side of any production.

Chomsky Hierarchy : Chomsky hierarchy classified the grammars into four parts.

The given diagram shows hierarchy of grammar and their accepting device. The production of above grammar are as follows :

(i) Regular $\Rightarrow A \to aB, A \to a, \ A, B \in V, a \in \Sigma$

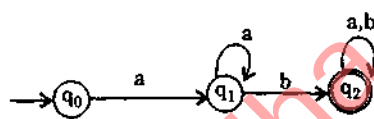
(ii) Context free $\Rightarrow A \to \alpha, \ A \in V, \alpha \in (V \cup \Sigma)^*$

(iii) Context sensitive $\Rightarrow A \to \alpha, \ \alpha \in (V \cup \Sigma)^*, \ A \in V$

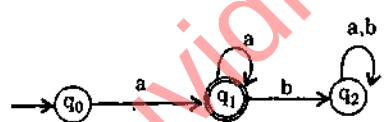
(ii) Type-0 $\Rightarrow \alpha \to B, \ \alpha, B \in (V \cup \Sigma)^*$

Q. 1. (d) Given a DFA M. Suggest a procedure to draw DFA which accepts the complement of the language accepted by M.

Ans. If L is a regular set over Σ then $\Sigma^* - L$ is also regular over Σ. We construct another DFA M' $(Q, \Sigma, \delta, q_0, F')$ by defining $F' = Q - F$, *i.e.*, M and M' differ only in their final states. A final state of M' is a nonfinal state of M and vice versa. The state diagram of M and M' are the same except for the final state.



Hence M' is



$W \in +(M')$ if and only if $\delta(q_0, W) \in F' = Q - F$

i.e., if $W \notin L$. This proves

$T(M') = \Sigma^* - X$

Q. 1. (e) Define equivalence relation and show that an equivalence relation partitions a set into a number of disjoint equivalence classes.

Ans. Assume that R is a relation on a set A in other word $R \subseteq A \times A$. As noted prev we write aRb instead of $(a, b) \in R$ to indicate that a is retated to b via R.

1. R is reflexive if for every $a \in A$ aRa
2. R is symmetric if for every a and b in A if aRb then bRA.
3. R is transitive if for every a, b and c in A if aRb and bRc then aRc.
4. R is an equivalence relation on A if R is reflexive, symmetric and transitive suppose R is an equivalence relation on A. For any element a of A, we denote by $[a]R$ or simply by $[a]$ the equivalence class containing a

$[a]R = \{x \in A \ / \ xRa\}$

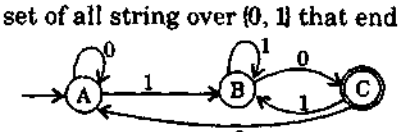
Note that the equivalence class containing a really does contain a.

For any partition C of a set A the relation R on A defined by.

xRy if and only if x and y belong to the same element of C is an equivalence relation on A.

Q. 1. (f) Give the format of production rules for left and right linear grammars and show that they are equivalent.

Ans. Let $L = \{0, 1\}^* \{10\}$ the set of all string over $\{0, 1\}$ that end in 10



$A \Rightarrow 1B \Rightarrow 11B \Rightarrow 110C \Rightarrow 1100A \Rightarrow 11000A$

$\Rightarrow 110001B \Rightarrow 1100010C \Rightarrow 11000101B$

$\Rightarrow 110001010C$

$A \to 1B \qquad C \to 0A$

$B \to 1B \qquad A \to 0A$

$B \to 0C \qquad C \to 1B$

These include every prod uction of form

$P \to aQ$ where $P \xrightarrow{a} Q$

is a transition in fA.

Q. 2. Attempt any four of the following :

(4 × 5 = 20)

Q. 2. (a) Write a regular expression for the language containing all strings of 0 and 1 that begin with 1 and contain even number of 0. Convert this regular expression into equivalent NFA without ε-moves.

Ans. $L = \{00, 01, 10, 11\}^*$

regular expression

$$1(00 + 01 + 10 + 11)^*$$

or $$1((0 + 1)(0 + 1))^*$$

Q. 2. (b) Define a regular expression and write various operators used in constructing a regular expression. Also give their precedence.

Ans. (i) $\phi + R = R$

(ii) $\phi R = R\phi = \phi$

(iii) $\wedge R = R\wedge = R$

(iv) $\wedge^* = \wedge$ and $q^* = \wedge$

(v) $R + R = R$

(vi) $R^* R^* = R^*$

(vii) $RR^* = R^* R$

(viii) $(R^*)^* = R^*$

(ix) $\wedge + RR^* = R^* = \wedge + R^* R$

(x) $(PQ)^* P = P(QP)^*$

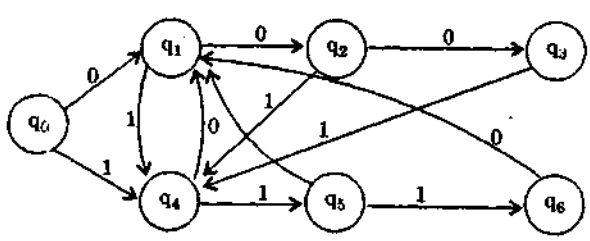
(xi) $(P + Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$

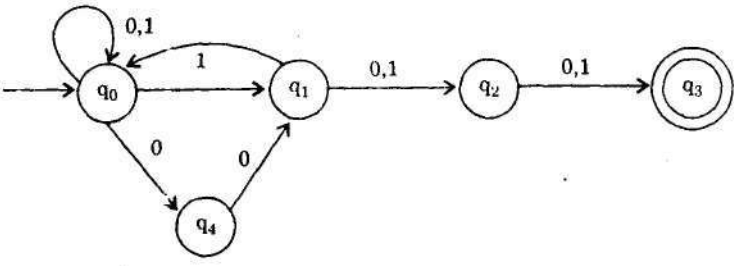
(xii) $(P + Q)R = PR + QR$ and

$R(P + Q) = RP = RQ$

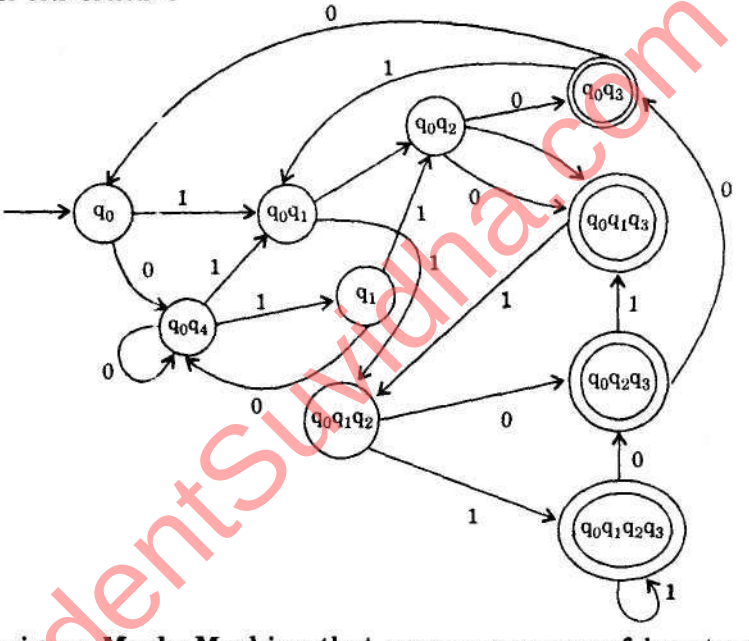
Q. 2. (c) Draw an NFA that accepts all strings of 0 and 1 containing 111 as substrings and hence convert this NFA to DFA.

Ans. The NFA is





The DFA after conversion :s



Q. 2. (d) Design a Mealy Machine that scans sequence of inputs of 0 and 1 and generates output 'A' if the input string terminates in 00, output 'B' if the string terminates in 11, and output 'C' otherwise.

Ans. Mealy Machine : If the string ends an 11 or 00 O/P = Yes

If the string not ends on otherwise O/P = No

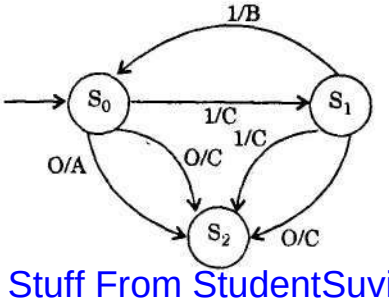
Set of states S $\{S_0, S_2, S_2\}$

Initial state $S = S_0$

Input alphabet E {0, 1}

Output alphabet Y $\{A, B, C\}$

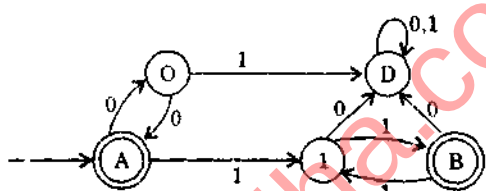
Input words can be any sequence of ones and zeros.



State	Input	New Scale	Output
S_0	0	S_2	C
S_0	1	S_1	C
S_1	0	S_2	C
S_1	1	S_0	B
S_2	0	S_0	A
S_2	1	S_1	C

Q. 2. (e) Draw a DFA that accepts the language $L = \{a^{2n} \mid n \geq 1\}$.

Ans. $\{00\}^* \{11\}^*$



Q. 2. (f) Write the statement of Pumping Lemma and using it prove that the language $\{a^n b^n c^n \mid n \geq 1\}$ is not regular.

Ans. 1. Let us suppose L is regular long and we get a contradiction suppose finite automotor has n states that accepts long L.

2. Let $z = 0^n 1^n$ then $|z| = 2n > n$. By pumping Lemma are write $z = uvw$ with $|uv| \leq n$ and $|v| \neq 0$.

3. Now we have to find l so that $uv^l \in L$ for getting a contradiction.

The string V can be in any one of the tree possible to

1. The string V is constructed by using only O's it means $v = O^k$ for some $k \geq 1$.

2. The string v is constant by using only 1's it means $v = 1^l$ for some $l \geq 1$.

3. The string v is constructed by using both symbol 0's and 1's. Then the string v will be of the form $v = 0^m 1^p$ for some $m \geq 1$ and $p \geq 1$.

In case (1) we take $l = 0$. As $uvw = 0^n 1^n$

$uw = 0^{n-k} 1^n$ as $k \geq 1$, $n - k \neq n$. Therefore $uw \notin L$.

In case II we take $l = 0$ $uw = 0^n 1^{n-l}$, $n \neq n - l$ $uw \notin L$

In case III if we take $l = 2$

As we have

$uvw = 0^{n-m} 0^m 1^p 1^{n-p}$

Now uv^2w will be

$$0^{n-m} \underbrace{0^m 1^p 0^m 1^p}_{u \quad v^2 \quad w} 1^{n-p}$$

As the string $0^{n-m} 0^m 1^p 0^m 1^p 1^{n-p}$ can not be expresed in the form $0^n 1^n$ $\therefore uv^2w \notin L$.

Hence the long L is not regular long.

Q. 3. Attempt any two of the following :

$(2 \times 10 = 20)$

Q. 3. (a) Consider the grammar with following production rules :

S → AB | AC

A → aA | bAa | a

B → bbA | aB | AB

C → aCa | aD

D → aD | bC

Convert the above grammar into Greibach Normal Form.

Ans. Given production of $CFG(V_N, \Sigma, P, S)$ where production P are as follows :

S → AB | AC

A → aA b | bAa | a

B → bbA | aaB | AB

C → abCa | aDb

D → bD | aC

Construction of P' : Now we can change the production of A as : $A \rightarrow aab / baa / a$

Now, we can change the production of B as :

$B \rightarrow bba / aabb\ A / aB$

$B \rightarrow bba / aabba / a\ bb\ A.$

$B \rightarrow bba / aabba / a\ bb\ aa) / abbbaa / aa$

Now, we can make any change in variable C and D and it is also not terminated so it will not include in V_N.

Hence our final CFG is as follows :

$P \rightarrow$

$S \rightarrow AB / AC$

$A \rightarrow aab / baa / a$

$B \rightarrow bba / aabba / abbaai / abbbaa / aa.$

Q. 3. (b) By drawing two rightmost or two leftmost derivations for a suitably selected string, show that the following grammar is an ambiguous grammar

E → E + E | E – E | E * E | E / E | D

D → a | b | c | d

Also convert the grammar into un-ambiguous grammar.

Ans. For $E \rightarrow E + E$

New production $E \rightarrow E + T / T$

for $E \rightarrow E^* E$

New production $E \rightarrow E^* f / f$

$E \rightarrow a / b$ remain same because they do not show any ambiguity.

Hence the constructed unambigoues grammar (G') equivalent to G is

$G' = (\{E, T, f\}, \{a, b, c, d\}\ P\ E)$

$P = \{E \rightarrow E + T / T, E \rightarrow E^* f / f, E \rightarrow a / b\}$

Q. 3. (c) Use CYK algorithm to check whether string a + b * c – d is in L (G) of the un-ambiguous grammar obtained in (b) above.

Ans. The given string is $W = a + b * c - d$

the grammar is :

$E \rightarrow E + E \,|E - E|E^* E|E / E|E / D$

$D \rightarrow a / b / c / d$

Let E be S_1 then the grammar will be

$S_1 \rightarrow S_1 + S_1|S_1 * S_1|S_1 - S_1|S_1 / S_1|a|b|c|d$

For CYK algorithm, the grammer must be in CNF.

A grammar will be in CNF if it follows the following rules :

Variable ⟶ Terminar

or

Variable ⟶ Variable_1 Variable_2

A grammar will be in CNF only if it is a simplified grammer so, first of all we will make the given grammar simplified.

$\Rightarrow E \to E + E|E - E|E * E|E / E|a|b|c|d$

$\Rightarrow E \to E + T|E - R|E * S|E / P|a|b|c|d|$

$T \to$

$\vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots$

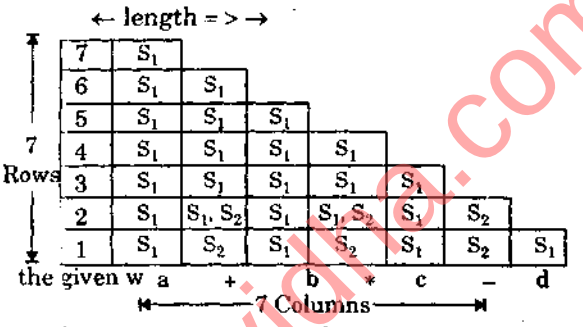
The GNF grammar will be:

Let S_2 be $S_2 \to +|*|=/$

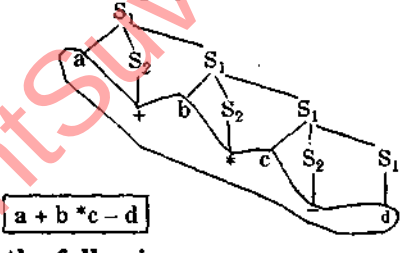
then $S_1 \to aS_2S_1|bS_2S_1|cS_2S_1|dS_2S_1$

$S_1 \to a|b|c|d,\ S_2 \to +|-|*|/\ W$

Now CYK : We have to check $W = a * b - c / d$



We will fill the table row by row moving upward.



Q. 4. Attempt any two of the following :

(2 × 10 = 20)

Q. 4. (a) Write transition rules for a PDA corresponding to the following Context Free Language :

$L = \{wcw^R \mid w$ is in $(0 + 1)^*$ and w^R represents reverse $W\}$

Also obtain context Free Grammar for this PDA.

Ans. $M = (\{q_0, q_1, q_2\}, \{0, 1, 2\}, \{0, 1, z_0\}, \delta_1, q_0, z_0, \{q_2\})$

$\delta(q_0, 0, z_0) = (q_0, 0z_0)$ $\quad \delta(q_0, 2, 0) = (q_1, 0)$

$\delta(q_0, 1, z_0) = (q_0, 1z_0)$ $\quad \delta(q_0, 2, 1) = (q_1, 1)$

$\delta(q_0, 0, 0) = (q_0, 00)$ $\quad \delta(q_0, 2, z_0) = (q_{11}z_0)$

$\delta(q_0, 1, 1) = (q_0, 11)$ $\quad \delta(q_1, 0, 0) = (q_1, \pi)$

$\delta(q_0, 1, 0) = (q_0, 10)$ $\quad \delta(q_1, 1, 1) = (q_1, \pi)$

$\delta(q_0, 0, 1) = (q_0, 0, 1)$ $\quad \delta(q_1, \pi, z_0) = \{q_0, z\}$

Q. 4. (b) Prove that the PDA that accepts strings through empty stack and final state mechanisms are equivalent.

Ans. $\{a^m b^m c^n \mid m, n \geq 1\}$

$\delta(q_0, a_1, z_0) = (q_0, az_0)$

$\delta(q_0, a, a) = (q_0, aa)$

$\delta(q_0, b, a) = (q_1, \in)$

$\delta(q, b, a) = (q, \in)$

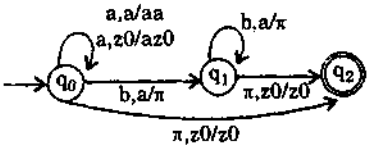
$\delta(q_1, c, z_0) = (q_1, z_0)$

$\delta(q_0, \in, z_0) = (q_f, \in)$

Q. 4. (c) Draw a PDA for the CFG given below :

$$S \rightarrow a \mid b \mid \varepsilon$$

Ans.

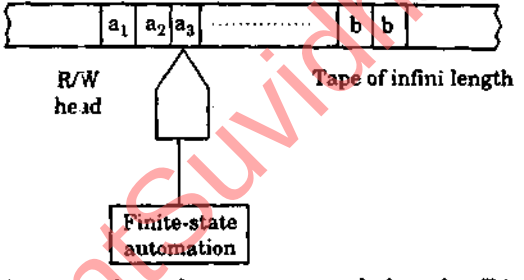


Q. 5. Attempt any four of the following :

(4 × 5 = 20)

Q. 5. (a) How do you define Instantaneous. Description for a Turning Machine ?

Ans. Turing Machine : The turing machine can be thought of as a finite state automaton connected to a R/W head the basic model for turing machine is given as follows :



In one move the machine examines the present symbol under R/w head on the tape and the present state of an automation to determine.

(i) a new symbol to be written on the tape.

(ii) motion of the R/W head either left (L) or right (R).

(iii) the next state.

(iv) whether to half or not

A turing machine M is a 7-tuple

i.e., $(Q, \Sigma, T, \delta, q_0, b, F)$

where

Q = is a finite non-empty set of states

T = is a finite non-empty set of tape symbols

$b \in T$ = is blank

Σ = is a set of input symbols and is a subset of T

δ = is transition function from $Q \times T$ to $Q \times T \times (L, R)$

q_0 = is initial state

F = is a set of final states.

Q. 5. (b) Design a Turing Machine that accepts the language $L = \{ww^R \mid w$ is in $(0 + 1)^*$ and w^R represents reverse $w\}$.

Ans. Turing machine for the language wwR | w is any string of 0's and 1's w^R is the reverse of string w. We have the following steps for processing ww^R.

(a) The Turing machine M scans the first symbol of the input tape (0 and 1), erase it and change state (q_1 or q_2).

(b) M scans the remaining part without changing the tape symbol until it encounters b.

(c) The R/w head moves to the left. If the rightmost symbol tallies with the leftmost symbol, the right most symbol is erased, other wise M halts.

(d) The R/w head moves to the left until b is encountered.

Transition Table

	Input symbol		
Present state	**0**	**1**	***n***
$\rightarrow q_0$	bRq_1	bRq_2	bRq_7
q_1	$0Rq_1$	$1Rq_1$	bLq_3
q_2	$0Rq_1$	$1Rq_2$	bLq_4
q_3	$0Lq_5$	–	–
q_4	–	bLq_5	–
q_5	$0Lq_5$	$1Lq_5$	bRq_0
q_6	$0Lq_6$	$1Lq_6$	bRq_0
(q_7)	–	–	–

Q. 5. (c) Write a Turing machine that performs proper subtraction of two integer numbers.

Ans. Assume that the input tape has 0^m10^n where $M - n$ is required. We have the following steps :

(a) the leftmost 0 is replaced by b and R/W head moves to the right.

(b) the R/W head replaces the first 0 after 1 by 1 and moves to the left. On reaching the biank at the left end the cycle is repeated.

(c) Once the 0's to the left of 1's are exhausted M replaces all 0's and 1's by b's $a - b$ is the number of 0's left over in the input tope and equal to 0.

(d) Once the 0's to the right of 1's are exhausted, n 0's have been changed to 1's by one 0 and n b's. The number of 0's left over gives the values of $a - b$ the transition table is given as follows :

Present state	**Input symbol**		
	0	1	b
$\rightarrow q_0$	bRq_1	bRq_5	–
q_1	$0Rq_1$	$1Rq_2$	–
q_2	$1Lq_3$	$1Rq_2$	bLq_4
q_3	$0Lq_3$	$1Lq_3$	bRq_0
q_3	$0Lq_4$	bLq_4	$0Rq_6$
q_5	bRq_5	bRq_5	bRq_6
q_6	–	–	–

Q. 5. (d) What is Post Correspondence Problem (PCP) ? Discuss it with one example.

Ans. The post correspondence problem over an alphabet Σ belongs to a class of Yes/No problems and is stated as follows :

consider the two lists $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_n)$ of nonempty strings over an alphabet $\Sigma = \{0, 1\}$. The PCP is to determine whether or not there exist $i_1 \ldots i_m$ where $1 \le i_j \le n$. Such that

$x_{i_1} x_{i_2} \ldots x_{i_m} = y_{i_1} y_{i_2} \ldots y_{i_m}$

The indices i_j's need not be distinct and m may be greater than n. Also if there exists a solution to PCP, there exists infinitely many solutions.

Post correspondence problem is undecidable : Let (T, W) be an arbitrary instance of Halting, so, that $T = (Q, \Sigma, \Gamma, q_0, \delta)$ is a turing machine and w is a string over the input alphabet Σ, we wish to construct an instance $(\alpha_1, \beta_1), (\alpha_2, \beta_2) \ldots (\alpha_n, \beta_n)$ of modified post correspondence problem (MPCP) that will have a solution if and only if T accepts W.

It is convenient to assume that T never crashes. Because there is an algorithm to convert any turing infinite loop whenever the roginal one crashes, we may make this assumption without loss of generality.

But the string W may be any thing. It may be generated by context free grammar or context sensitive grammar or type-0.

(i) If L_1 and L_2 are any two CFL over an alphabet, there is no algorithm to determine whether or not

(a) $L_1 \cap L_2 = \phi$

(b) $L_1 \cap L_2$ is a CFL

(c) $L_1 \subseteq L_2$

(d) $L_1 = L_2$

(ii) If G is a context sensitive grammar, there is no algorithm to determine whether or not

(a) $L(G) = \phi$

(b) $L(G)$ is infinite

(c) $x_0 \in L(G)$ for a fixed string no.

(iii) If G is a type-0 grammar, there is no algorithm to determine whether or not any string $x \varepsilon \Sigma^*$ is in $L(G)$.

Hence we can say that post correspondence problem is undecidable, because there is no any algorithm to decide that TM will halt or not.

Q. 5. (e) Write a short note on tne Multi Tape Turing Machine.

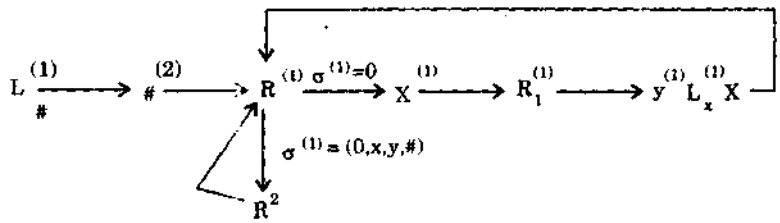
Ans. At the begining

First tape # 0 0 1 1 1 #

Second tape #

The movement of head on first tape is

#	0	0	1	1	1	#
#	0	0	1	1	1	#
#	X	0	1	1	1	#
#	X	0	1	1	1	#
#	X	0	Y	1	1	#
#	X	0	Y	1	1	#
#	X	0	Y	1	1	#
#	X	X	Y	1	1	#
#	X	X	Y	1	1	#
#	X	X	Y	Y	1	#
#	X	X	Y	Y	1	#
#	X	X	Y	Y	1	#
#	X	X	Y	Y	1	#
#	X	X	Y	Y	1	#



This one will store on the second tape.

Q. 5. (f) Differentiate with suitable examples between recursive and recursively enumerable languages.

Ans. Suppose that $T_1 = (Q_1, \Sigma, \Gamma_1, q_1, \delta_1)$ and $T_2 = (Q_2, \Sigma, \Gamma_2, q_2, \delta_2)$ are two turing machine accepting L and L' respectively is compliment of L.

We construct a two-tape $TM \rightarrow T$ to recognize L.

We include in our set of states the pairs in $Q_1 \times Q_2$.

The two tape machine $T = (Q, \Sigma, \Gamma, q_0, \delta)$ begins by copying the input string x onto tape z and inserting the market # at the beginning of each tape. The simultaneous simulation on T_1 on tape 1 and tape 2 is accomlished by allowing every possible more

$\delta((P_1, P_2), (a_1, a_2)) = ((q_1, q_2), (b_1, b_2), (D_1, D_2))$

where for both values of i

$\delta_i(pi, ai) = (qi, bi, Di)$

Here we know in advance that on any input x, precisely one of the machine T_1 and T_2 will halt.

Therefore it is sufficient to modify T as follows :

(i) When T_1 or T_2 halts, T erases tape 1 and

(ii) leaves the appropriate output before halting-1 or 0 depending on whether the machine that halted was the one accepting L or the one accepting L.