Unit-1

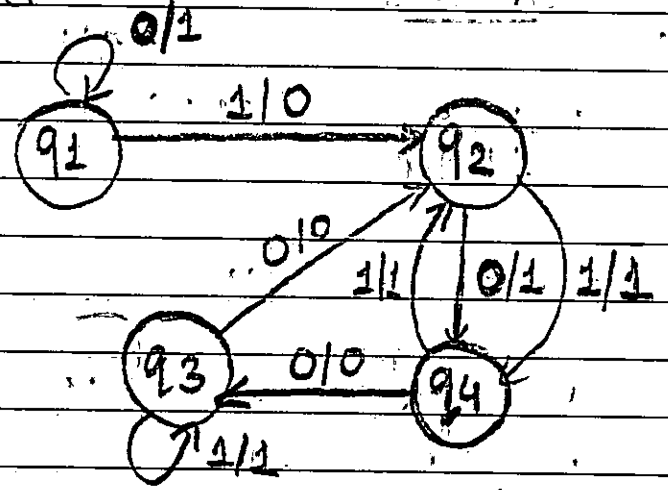
Q1- (a)

	DFA	NFA
(i)	DFA stands for deterministic finite automata.	NFA stands for non-deterministic finite automata.
(ii)	A finite automata is said to be deterministic if corresponding to an input symbol there is only one transition.	A finite automata is said to be non-deterministic if corresponding to an input symbol there are more then one transition.
(iii)	ϵ-moves are not allowed.	ϵ-moves are allowed in NFA.
(iv)	Digital computers are associated with DFA.	NFA is not familiar with the real world computers.
(v)	$\delta : Q \times \Sigma = Q$ Ex:- $q_0 \xrightarrow{a} q_1$	$\delta : Q \times \Sigma = 2^Q$ Ex:- $q_0 \xrightarrow{a} q_0$, $q_0 \xrightarrow{a} q_1$

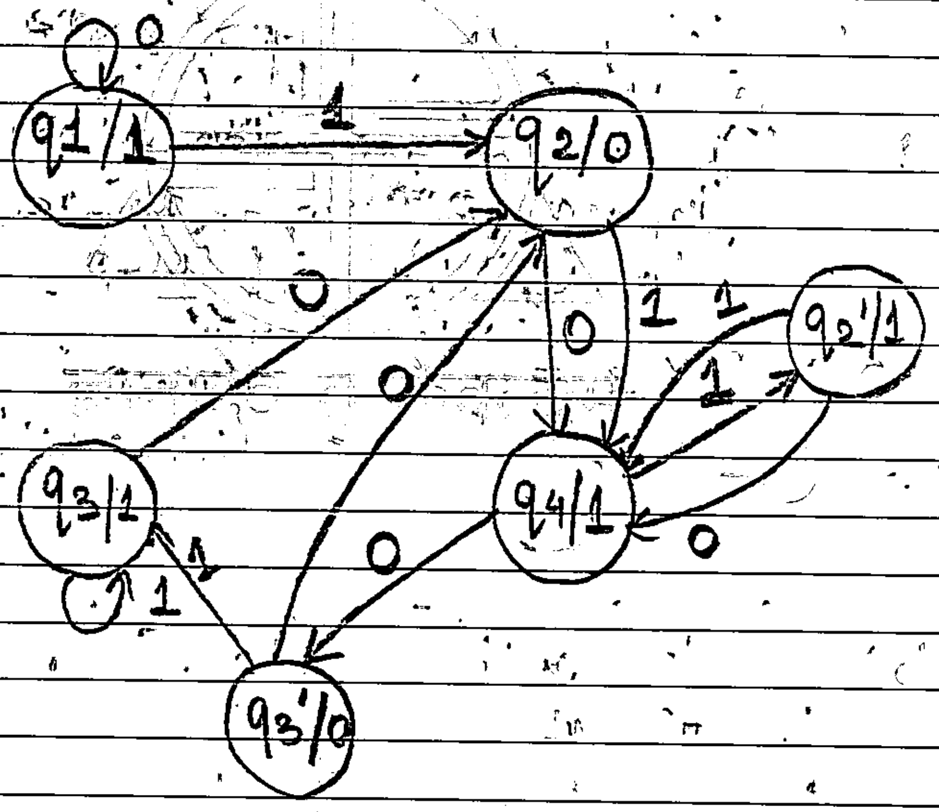
Q1- (b) Mealy Machine:- Transition Table.

	a = 0 State	a = 0 O/P	a = 1 State	a = 1 O/P
q_1	q_1	1	q_2	0
q_2	q_4	1	q_4	1
q_3	q_2	0	q_3	1
q_4	q_3	0	q_2	1

Mealy Machine:-



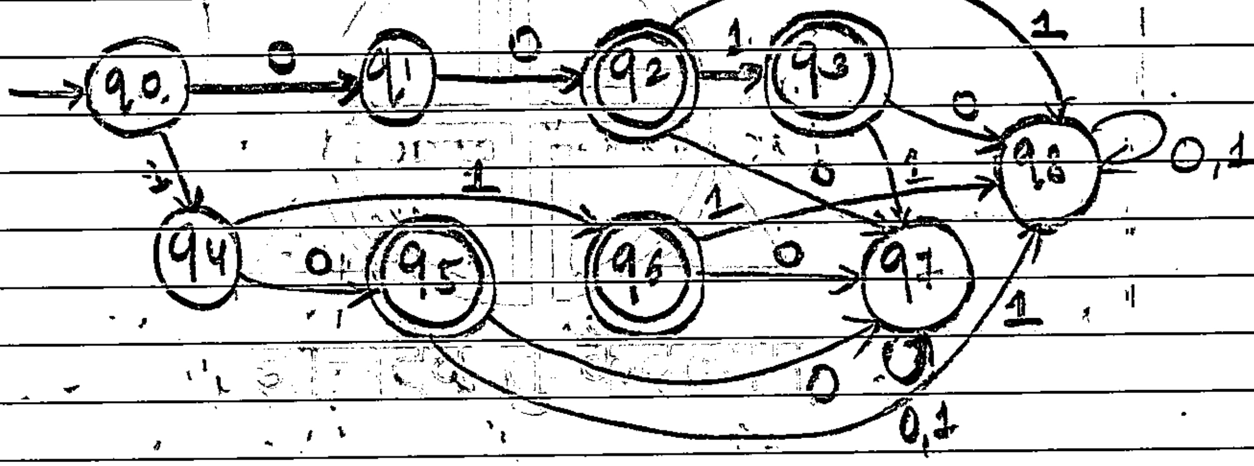
Moore Machine:-



Transition Table for Moore Machine :-

Current State	Next State 0	Next State 1	Output
q1	q1	q2	1
q2	q4	q4	0
q2'	q4	q4	1
q3	q2	q3	1
q3'	q2	q3	0
q4	q3'	q2'	1

Q1(C) :-



	q0	q1	q2	q3	q4	q5	q6	q7	q8
q0									
q1	✓								
q2	✓	✓							
q3	✓	✓							
q4	✓		✓	✓					
q5	✓	✓			✓				
q6	✓	✓			✓				
q7	✓	✓	✓	✓	✓	✓	✓		
q8	✓	✓	✓	✓	✓	✓	✓		

All the unmarked pairs are :- q_0q_1 [1], q_2q_3 [2], q_0q_4 [3], q_1q_4 [4], q_2q_5 [5], q_3q_5 [6], q_2q_6 [7], q_3q_6 [8], q_5q_6 [9], q_0q_7 [10], q_1q_7 [11], q_4q_7 [12], q_0q_8 [13], q_1q_8 [14], q_4q_8 [15], q_7q_8 [16]

(1) $q_0q_1 \Rightarrow \delta(q_0,0) = q_1 \quad \delta(q_0,1) = q_4$ ✓ matched.
$\delta(q_1,0) = q_2 \quad \delta(q_1,1) = q_3$ (q_1 & q_2 are ticked)

(2) $q_2q_3 \Rightarrow \delta(q_2,0) = q_7 \quad \delta(q_2,1) = q_8$ ✗
$\delta(q_3,0) = q_8 \quad \delta(q_3,1) = q_7$

(3) $q_0q_4 = \delta(q_0,0) = q_1 \quad \delta(q_0,1) = q_4$ ✓
$\delta(q_4,0) = q_5 \quad \delta(q_4,1) = q_6$

(4) $q_1q_4 = \delta(q_1,0) = q_2 \quad \delta(q_1,1) = q_3$ ✗
$\delta(q_4,0) = q_5 \quad \delta(q_4,1) = q_6$

(5) $q_2q_5 = \delta(q_2,0) = q_7 \quad \delta(q_2,1) = q_8$ ✗
$\delta(q_5,0) = q_7 \quad \delta(q_5,1) = q_8$

(6) $q_3q_5 = \delta(q_3,0) = q_8 \quad \delta(q_3,1) = q_7$ ✗
$\delta(q_5,0) = q_7 \quad \delta(q_5,1) = q_8$

(7) $q_2q_6 = \delta(q_2,0) = q_7 \quad \delta(q_2,1) = q_8$ ✗ (continue)
$\delta(q_6,0) = q_7 \quad \delta(q_6,1) = q_8$

(8) $q_3q_6 = \delta(q_3,0) = q_8 \quad \delta(q_3,1) = q_7$ ✗
$\delta(q_6,0) = q_7 \quad \delta(q_6,1) = q_8$

(9) $q_5q_6 = \delta(q_5,0) = q_7 \quad \delta(q_5,1) = q_8$ ✗
$\delta(q_6,0) = q_7 \quad \delta(q_6,1) = q_8$

(10) $q_0q_7 = \delta(q_0,0) = q_1 \quad \delta(q_0,1) = q_4$ ✗ (q_0q_7) ✓ matched
$\delta(q_7,0) = q_7 \quad \delta(q_7,0) = q_7$

(11) $q_1q_7 = \delta(q_1,0) = q_2 \quad \delta(q_1,1) = q_3$ ✓
$\delta(q_7,0) = q_7 \quad \delta(q_7,1) = q_7$

(12) $q_4q_7 = \delta(q_4,0) = q_5 \quad \delta(q_4,0) = q_6$ ✓
$\delta(q_7,0) = q_7 \quad \delta(q_7,1) = q_7$

(13) $q_0q_8 = \delta(q_0,0) = q_1 \quad \delta(q_0,1) = q_4$ ✗ (q_0q_8) ✓
$\delta(q_8,0) = q_8 \quad \delta(q_8,1) = q_8$ {as q_1 & q_8 are matched now}

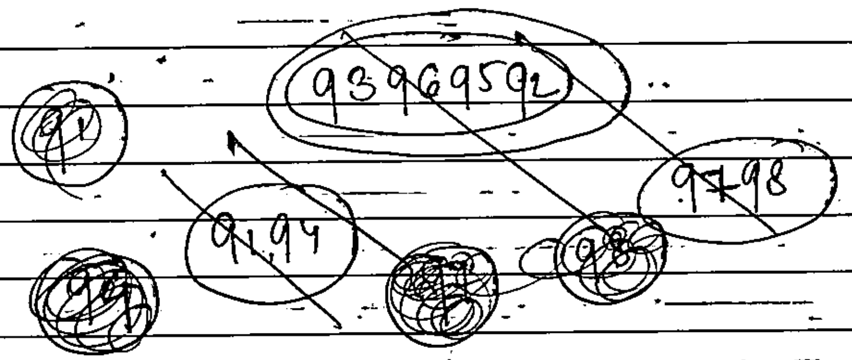
Q2 (a) Closure property of regular grammer :-
(i) Union :- If two languagis are regular then there union will also be regular.
(ii) Cantatenation — (iv) Substitution
(iii) Homomorphism
Closure property generally defines that all the properties must satisfy that it is also a regular grammer.

(14). q_1q_8 — $\delta(q_1,0) = q_2$ $\delta(q_1,1) = q_3$ ✓
$\delta(q_8,0) = q_8$ $\delta(q_8,1) = q_8$

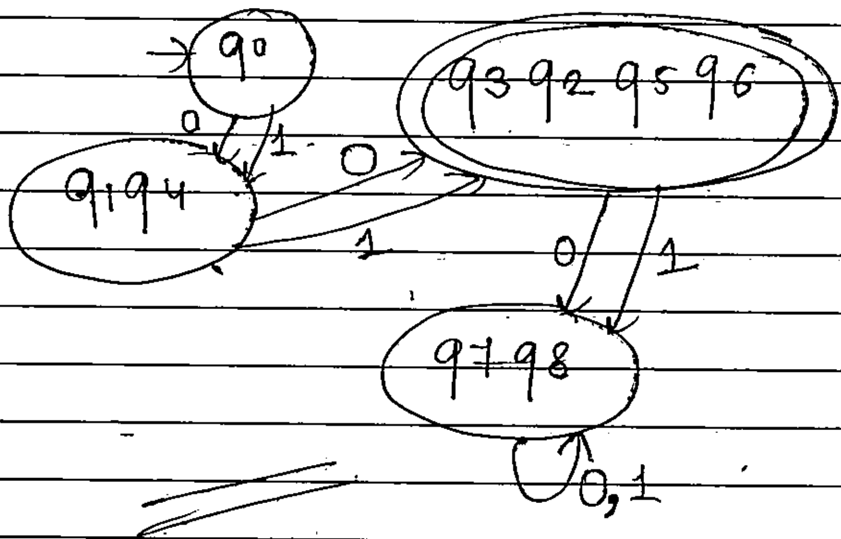
(18) q_4q_8 — $\delta(q_4,0) = q_5$ $\delta(q_4,1) = q_6$ ✓
$\delta(q_8,0) = q_8$ $\delta(q_8,1) = q_8$

(16). q_7q_8 — $\delta(q_7,0) = q_7$ $\delta(q_7,1) = q_7$ ✗
$\delta(q_8,0) = q_8$ $\delta(q_8,1) = q_8$

unmatched pairs! — q_2q_3, q_1q_4, q_2q_5, q_3q_5, q_2q_6, q_3q_6, q_5q_6, q_7q_8.



q_2q_3
q_2q_6
q_2q_5
q_1q_4
q_3q_6
q_3q_5
q_3q_6
q_7q_8



Hence minimized DFA

Q2- (a) (backpage)

Q2-(c) For proving whether the language is regular or not we use pumping lemma. Pumping lemma is used to prove that the lang. is not regular.

If A is a regular langn. then it has a Pumping length 'P' such that S is the string where S ≥ P and s = xyz such that it satisfy the following conditions:-

1. $xy^iz \in A$
2. $|y| > 0$
3. $|xy| \leq P$.

$L = \{a^n b^n \text{ where } n \geq 1\}$.

Pumping length $P = 4$

$a^P b^P = a^4 b^4$

aaaabbbb

(1) Case 1:- $x = aa$, $y = aa$, $z = bbbb$.
xy^iz = where $i=1$. $\Rightarrow xy^1z$.
$\Rightarrow$ aaaabbbb.
when $i=2$.
$\Rightarrow$ aaaaaabbbb. X.

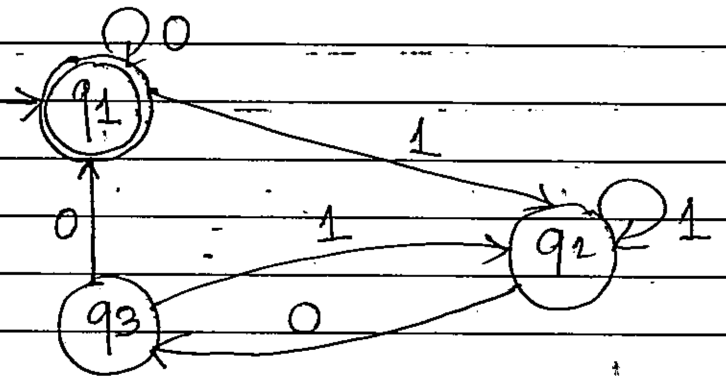
Case 2:- $x = aaaa$, $y = bb$, $z = bb$.
xy^iz when $i=1$
$\Rightarrow$ aaaabbbb.
xy^iz when $i=2$ $\Rightarrow xy^2z$.
$\Rightarrow$ aaaabbbbbb. X.
∵ length is not same. It is not true.
Case 3:- $x = aa$, $y = aabb$, $z = bb$.
xy^iz when $i=1$.
$\Rightarrow$ aaaabbbb
xy^iz when $i=2$.
$\Rightarrow$ aaaabbaabbbb X.
It is not true.

(2). $|y| > 0$.
$2 > 0$ (True). (when $y = aa$ or bb).

(3):- $|xy| \le p$.
$|4| \le p$ (True).

It is not a regular language as it is not satisfying following the condition.
Hence proved language is not regular.

Q2 - (d.)



first we will take final state:-

$$q_1 = \epsilon + q_1 0 + q_3 0 \quad \text{——} \quad (1)$$

$$q_2 = q_2 1 + q_1 1 + q_3 1 \quad \text{——} \quad (2)$$

$$q_3 = q_2 0 \quad \text{——} \quad (3)$$

Arden's Theorem
$R = Q + RP$
$R = QP^*$

$$q_1 = \epsilon + q_1 0 + q_3 0$$

$$q_2 = q_3 1 + \cancel{q_2 1} + \cancel{q_1 1}$$
$$= q_3 1 + \cancel{1 (0}$$

Putting value of q_3 in eqn (1).

$$q_1 = \epsilon + q_1 0 + (q_2 0) 0 \quad \text{——} \quad (4)$$

$$\cancel{q_1 = \epsilon + \epsilon 0 (q_1 + q_2)}$$

$$q_2 = q_2 1 + q_1 1 + q_3 1$$
$$q_2 = q_2 1 + q_1 1 + (q_2 0) 1$$
{ putting value of q_3 in eqn (2) }

$$\underset{R}{q_2} = \underset{Q}{q_1 1} + \underset{R}{q_2} (\underset{P}{1 + 01})$$

$$q_2 = q_1 1(1+01)^* \quad \text{--- } (5)$$

putting value of eqn (5) in eqn (4).

$$q_1 = \epsilon + q_1 0 + q_1 1(1+01)^* 00.$$

$$q_1 = (0+1)((1+01)^* 00)^*$$

Q3- (a)(1) $(0+1)^*1$

(2) $(0+1)^*00$

Q3- (b) Chomsky Classification of Grammers—

Chomsky introduced grammer and divided it into four types of grammer :—

(i) Type-0 grammer :- The grammer which comes under type-0 is unrestricted grammer which is recognized by Turing Machine. Turing Machine accepts or generate Type-0 grammer. Turing Machine is a model used in computation. Recursive enumerable language is a Type-0 language which are generated by Type-0 grammer.

(2) Type-1 grammer :- The grammer which comes under type-1 is context sensitive grammer which is accepted by linear bounded automata.

(3) Type-2 grammer:- The grammer which comes under type-2 is context free grammer which is accepted by PushDown automata. Context free languages comes from context free grammer and we use pushdown automata.

(4) Type-3 grammer:- The grammer which comes under type-3 is regular grammer which is accepted by finite automata. Regular ~~gra~~ expressions are the method used to represent regular languages and use finite automata for this.

Chomsky classification or hierarchy includes these 4 types and are further used in theory of computations to solve the problems.

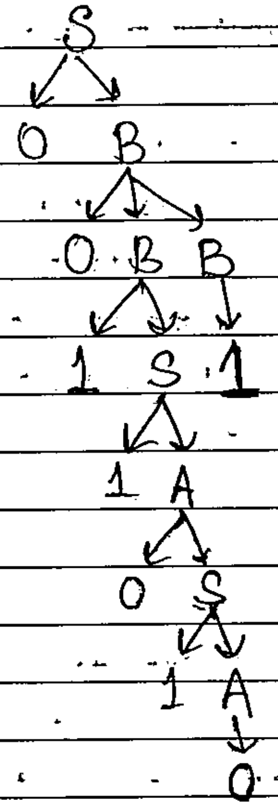
Q3-(d) $S \rightarrow 0B \mid 1A$

$A \rightarrow 0 \mid 0S \mid 1AA$

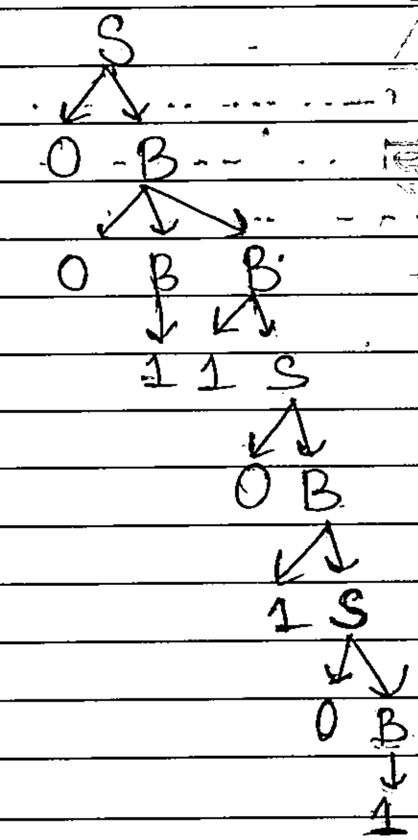
$B \rightarrow 1 \mid 1S \mid 0BB$

$w = 00110101$

LMD :- (Left Most Derivative tree)



RMD (Right Most Derivative Tree)



Hence generated LMD & RMD for the given String.

$w = 0011\,0101$

Q4- (a)

NPDA	DPDA
(1) It stands for non-deterministic push down automata.	(1) It stands for deterministic push down automata.
(2) Movable string are acceptable.	(2) Movable strings are not acceptable.
(3) Dead configuration is allowed.	(3) Dead configuration is not allowed.
(4) It is easy as compared to DPDA.	(4) It is difficult in understanding.
(5) It is not mostly used.	(5) It is mostly used.

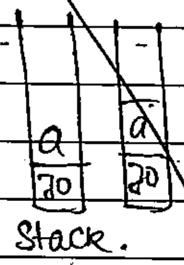
Q4- (b) $L = \{a^n b^{2n}\}$ where $n \geq 1$

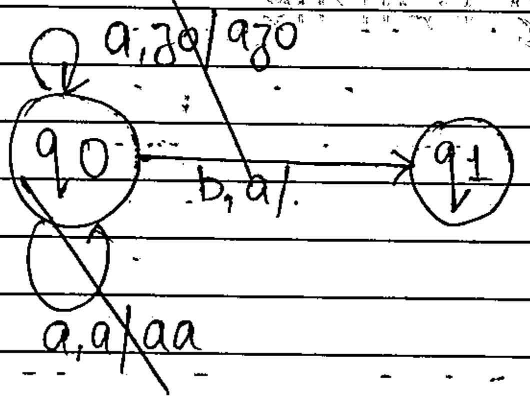
$a^n b^{2n}$

taking $n = 2$.

$a^2 b^4$

| a | a | b | b | b | b | ε. |

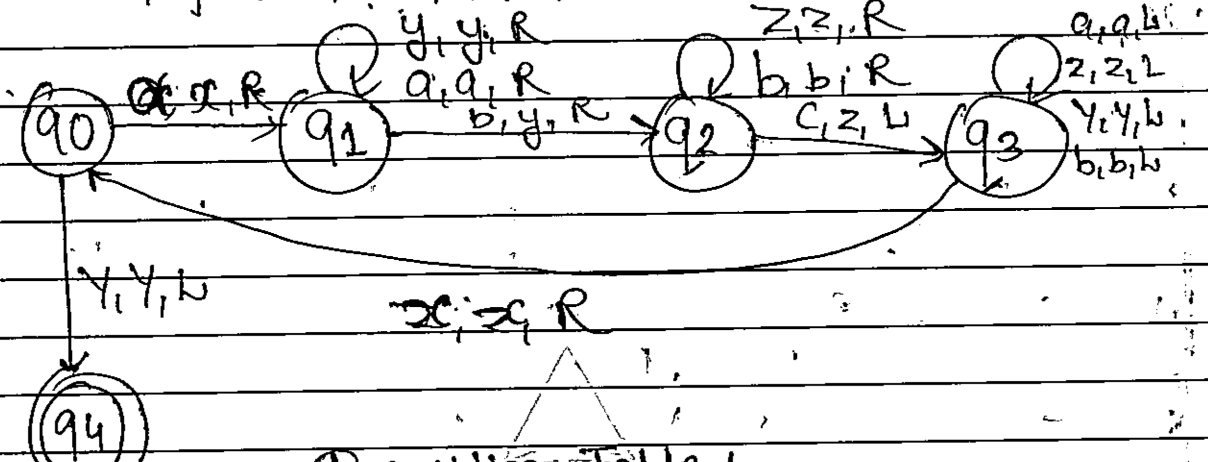




Qu- (c) $L = \{a^n b^n c^n\}$ where $n \geq 0$.
$a^3 b^3 c^3$.

$n = 3$,

B	B	a	a	a	b	b	b	c	c	c	B	B



Transition Table :-

$\delta(q0, a) \rightarrow (q1, xR)$
$\delta(q1, a) \rightarrow (q1, aR)$
$\delta(q1, b) \rightarrow (q2, yR)$
$\delta(q2, b) \rightarrow (q2, bR)$
$\delta(q2, z) \rightarrow (q2, zR)$
$\delta(q2, b) \rightarrow (q2, bR)$
$\delta(q2, c) \rightarrow (q2, zL)$
$\delta(q3, a) \rightarrow (q3, aL)$
$\delta(q3, z) \rightarrow (q3, z, L)$
$\delta(q3, y) \rightarrow \delta(q3, yL)$
$\delta(q3, b) \rightarrow \delta(q3, bL)$
$\delta(q3, x) \rightarrow \delta(q0, xR)$

$\delta(q0, y) \rightarrow \delta(q4, yL)$

Q4:- (d) (i) Decidable Problem :- Decidable problems are the problems in which ~~has~~ the Turing Machine halts after a finite amount of time to give the answer as 'yes' or 'no'. Decidable Problems comes under type-0. Recursive enumerable languages comes under decidable problems.

Example of decidable problems:- Equivalence of two regular languages.

Undecidable Problems :- Undecidable problems are the problems in which there is no turing machine which will halt in a given finite time for giving answer as 'yes' or 'no'. undecidable problems are the problems which don't have any algorithms to solve the problem.

Example of undecidable problem :- Ambiguity in context free language.

PCP is an undecidable decision problem.

(2) Halting Problem of Turing Machine:-

Given algorithms are 'yes' or 'no'? Halting means when an input symbol is accepted ~~by~~ ~~...~~ ~~...~~ then the machine halts or if it is rejected then it halts. Halting simply means terminating. For understanding halting we must know some of the terms of computation which includes:-

(1) Computability:- It is the ability to solve the problem in an appropriate manner. Turing machine generally halts when it accepts or rejects the string. Halting is an important step in Turing machine for terminating the string.

Q5- (a):- Initial function over N can be defined as:-

(1) zero function defined as $Z(x)=0$

(2) successor function defined by:
$S(x) = (x+1)$

(3) Projection function $U_i^n(x_1, \ldots, x_n)$.
defined by
$$U_i^n = x_i$$
Initial function over Σ is defined by
(1) $nil(x) = \Lambda$.
(2) $cons\,a(x) = ax$
(3) $cons\,b(x) = bx$.

Partial recursive function are the functions in which turing machine halts when accepted and may or may not halt if rejected.
Ex:- Substraction of two integers.

Q5-(b):- $f(x,y) = x^* y$
(1) taking $y=0$
$f(x,0) = x^* 0$
$= Z(x) = g(x)$
(2) taking $y = y+1$
$f(x,y+1) = x^*(y+1)$
$= x^* y + x^*$
(3) taking a new function.
$$h(x,y,f(x,y)) = U_3^3(x,y,f(x,y)) + U_1^3(x,y,f(x,y)).$$
taking $f(x,y)$ as z

$h(x,y,z) = U_3^3(x,y,z) * U_1^3(x,y,z)$

As $*$ is already a primitive recursive function, so we have proved $f(x,y) = x*y$ is also primitive recursive function.

② $f(x,y)\ x^y$

taking $y = 0$.

$f(x,y) = x^0 = 1$
$= g(x)$

taking $y = y+1$

$= x^{y+1}$

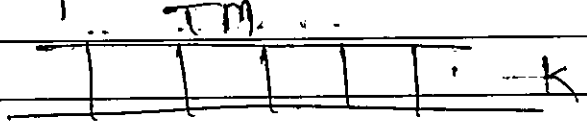
$f(x,y) = x^y . x$. taking new function 'h',

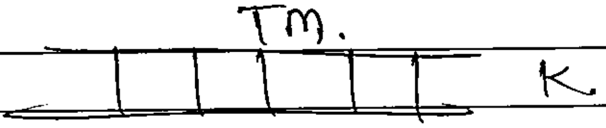
$h(x, y, f(x,y)) = U_3^3(x, y, f(x,y)) * U_1^3(x, y, f(x,y))$

As $*$ is already a primitive recursive function so we proved that

$f(x,y) = x^y$ is also a primitive recursive function.

Q5 - (c) Space and Time complexity:-





Time Complexity:- For a turning machine TM with K no of tapes, if the length of the string is 'n' then the turing machine can make atmost 'n' number of moves. This is the the time complexity of Turing machine which tells it covers in finite times.

$$\boxed{T(n) = O(n)}$$

Space Complexity:- For a tape of length 'n' the total no. of cells will always be 'n' that means for 'n' number of strings there are 'n' no. of cells available not more than that.

$$\boxed{S(n) = O(n)}$$

Time and space complexity are two important terms in theory of computation which makes it more efficient.

ज्ञानादेव तु कैवल्यम्

ज्ञानादेव तु कैवल्यम्

ज्ञानादेव तु कैवल्यम्

ज्ञानादेव तु कैवल्यम्

ज्ञानादेव तु कैवल्यम्

ज्ञानादेव तु कैवल्यम्

ज्ञानादेव तु कैवल्यम्