Paper / Subject Code: 31904 / Theory of Computer Science

Time: 3 Hours

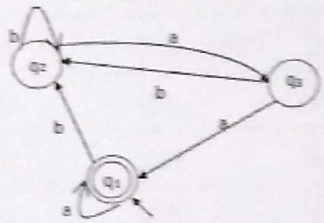
Total Marks: 80

N.B.: (1) Question No.1 is compulsory.
(2) Attempt any three questions from the remaining five questions.
(3) Make suitable assumptions wherever necessary but justify your assumptions.

1. (a) Differentiate DFA and NFA. 05
(b) Design a DFA to accept string of 0's and 1's ending with the string 100. 05
(c) Explain the applications of Regular Expressions. 05
(d) What are Recursive and Recursively Enumerable Languages? 05

2. (a) Design NFA for recognizing the strings that end in "aa" over $\Sigma = \{a,b\}$ & convert above NFA to DFA. 10
(b) Design moore m/c for following:- 10
If input ends in '101' then output should be A, if input ends in '110' output should be B, otherwise output should be C and convert it into mealy m/c.

3. (a) Obtain a regular expression for the FA shown below: 10



(b) Explain the types of Turing machine in detail. 10

4. (a) Design a turing machine that computes a function f(m,n)=m+n i.e. addition of two integers. 10
(b) State and explain pumping Lemma for Context Free Languages. Find out whether the language $L = \{x^n y^n z^n \mid n \geq 1\}$ is context free or not. 10

5. (a) Design PDA for the following language: 10

$L(M) = \{wcw^R \mid w \ \{a,b\}^*\}$ where w^R is reverse of w & c is a constant.
(b) Convert the following Grammars to the Chomsky normal form (CNF). 10

$S \rightarrow 0A0 \mid 1B1 \mid BB$
$A \rightarrow C$
$B \rightarrow S \mid A$
$C \rightarrow S \mid \varepsilon$

6. Write detailed note on **(any two):-** 20
(a) Post Correspondence Problem
(b) Halting Problem.
(c) Rice's Theorem.

****

68476

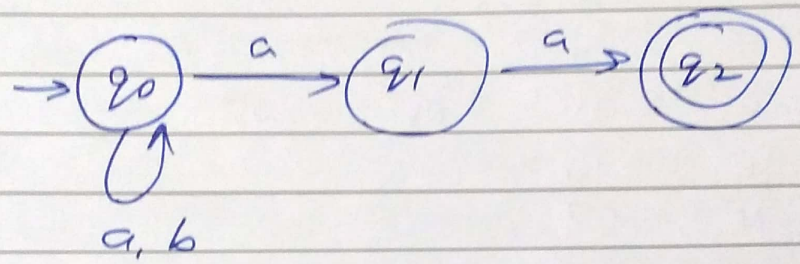
88A89FBA80932F83D9895BA53BF28E0D

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

2a) Design NFA for recognizing the strings that end in "aa" over $\Sigma = \{a, b\}$ & convert above NFA to DFA.

Ans NFA for recognizing strings over $\{a, b\}$ that end in "aa".



$M = (\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\})$

$\delta : Q \times \Sigma \cup \{\epsilon\} \rightarrow 2^Q$

Q \ $\Sigma \cup \{\epsilon\}$	a	b
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	ϕ
q_2	ϕ	ϕ

Converting above NFA to DFA.

Equivalent DFA $M' = (Q', \Sigma, \delta', q_0', F')$

where

$Q' = 2^Q = \{\phi, [q_0], [q_1], [q_2], [q_0, q_1], [q_0, q_2], [q_1, q_2], [q_0, q_1, q_2]\}$

rks of n no.		Examiner	
		Moderator	
		Re-Assessor	

Question No.	START WRITING HERE

$q_0' = [q_0]$

$F' = \{ [q_2], [q_0, q_2], [q_1, q_2], [q_0, q_1, q_2] \}$

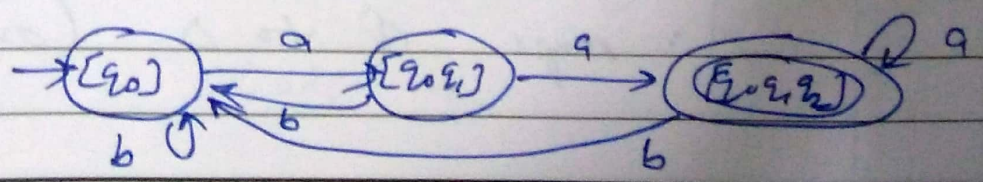
δ' maps $Q' \times \Sigma \to Q'$ as

Q' \ Σ	a	b
ϕ	ϕ	ϕ
$[q_0]$	$[q_0, q_1]$	$[q_0]$
$[q_1]$	$[q_2]$	ϕ
$[q_2]$	ϕ	ϕ
$[q_0 q_1]$	$[q_0, q_1 q_2]$	$[q_0]$
$[q_0 q_2]$	$[q_0, q_1]$	$[q_0]$
$[q_1 q_2]$	$[q_2]$	ϕ
$[q_0 q_1 q_2]$	$[q_0, q_1, q_2]$	$[q_0]$

Minimization of DFA gives

δ'' as

Q'' \ Σ	a	b
Start state $[q_0]$	$[q_0 q_1]$	$[q_0]$
$[q_0 q_1]$	$[q_0 q_1 q_2]$	$[q_0]$
* $[q_0 q_1 q_2]$	$[q_0 q_1 q_2]$	$[q_0]$



Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

2b) Design moore m/c for following :-
If input ends in '101' then output be A,
If input ends in '110' output should be B,
otherwise output should be C
and convert it into mealy m/c.

Ans MOORE M/c

$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$

where Q : finite set of states are

q_0 : Start state
q_1 : input ends with '1'
q_2 : input ends with '10'
q_3 : input ends with '11'
q_4 : input ends with '101'
q_5 : input ends with '110'

Σ : finite set of input symbols are
$\{0, 1\}$

Δ : finite set of output symbols are
$\{A, B, C\}$

δ : maps $Q \times \Sigma$ to Q (Transition function)

λ : maps Q to Δ (output function)

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

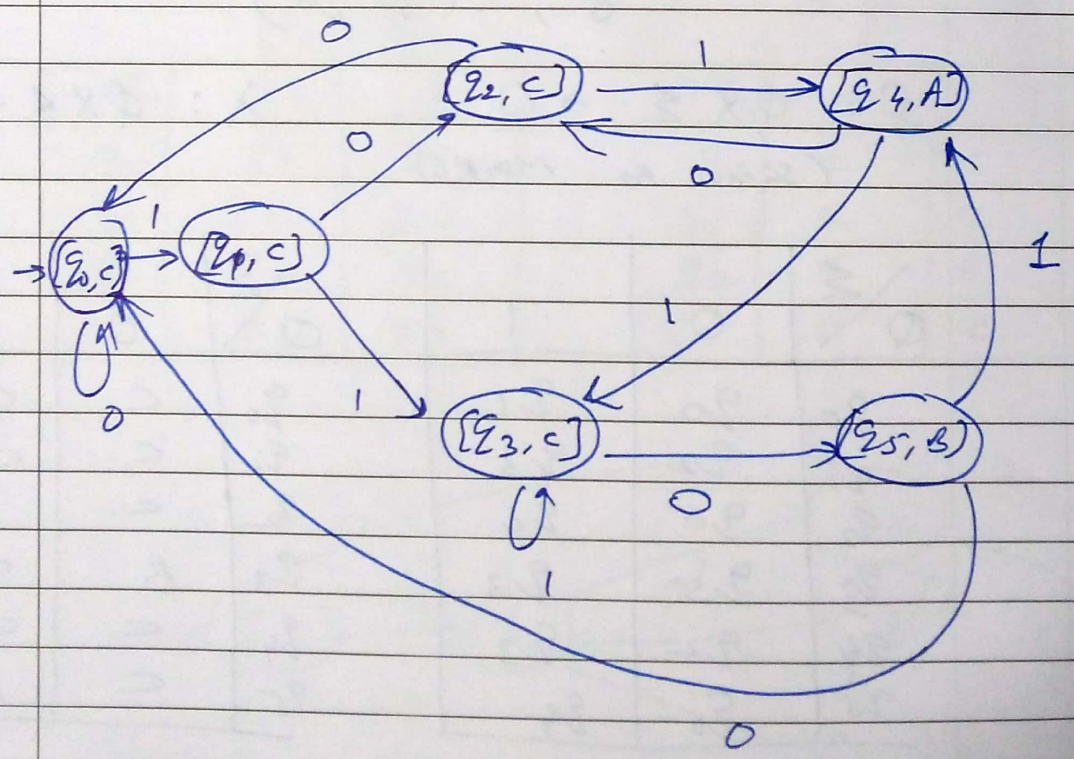
They are as follows:

$\delta : Q \times \Sigma \to Q$ $\qquad$ $\lambda : Q \to \Delta$

$Q \backslash \Sigma$	0	1
q_0	q_0	q_1
q_1	q_2	q_3
q_2	q_0	q_4
q_3	q_5	q_3
q_4	q_2	q_3
q_5	q_0	q_4

Q	Δ
q_0	C
q_1	C
q_2	C
q_3	C
q_4	A
q_5	B

Graphical representation



MOORE M/c

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

Converting above MOORE to MEALY as

~~$M' = (Q^{\Sigma}, \Sigma, \Delta, \delta', q_0$~~

$M' = (Q, \Sigma, \Delta, \delta, \lambda', q_0)$

where λ' maps $Q \times \Sigma \to \Delta$ as

$$\lambda'(p, a) = \lambda(\delta(p, a))$$

Mealy m/c will be

$M = (\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{0, 1\}, \{A, B, C\}, \delta, \lambda, q_0)$

$\delta : Q \times \Sigma \to Q$ (same as MOORE)

$\lambda : Q \times \Sigma \to \Delta$

Q \ Σ	0	1
q_0	q_0	q_1
q_1	q_2	q_3
q_2	q_0	q_4
q_3	q_5	q_3
q_4	q_2	q_3
q_5	q_0	q_4

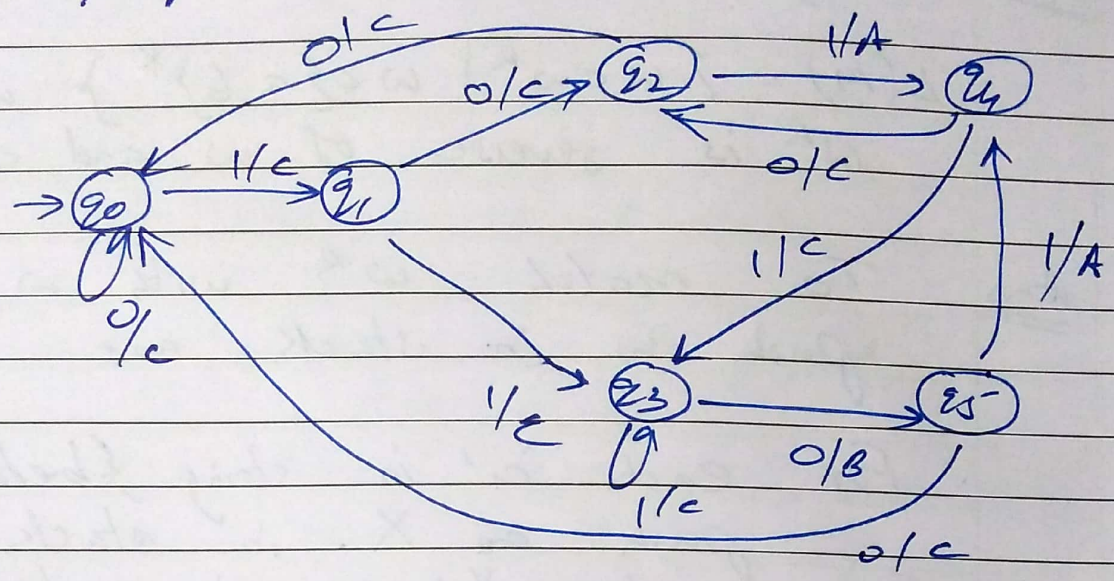
Q \ Σ	0	1
q_0	C	C
q_1	C	C
q_2	C	A
q_3	B	C
q_4	C	C
q_5	C	A

		Examiner	
s of no.		Moderator	
		Re-Assessor	

Question No.

START WRITING HERE

Graphical representation

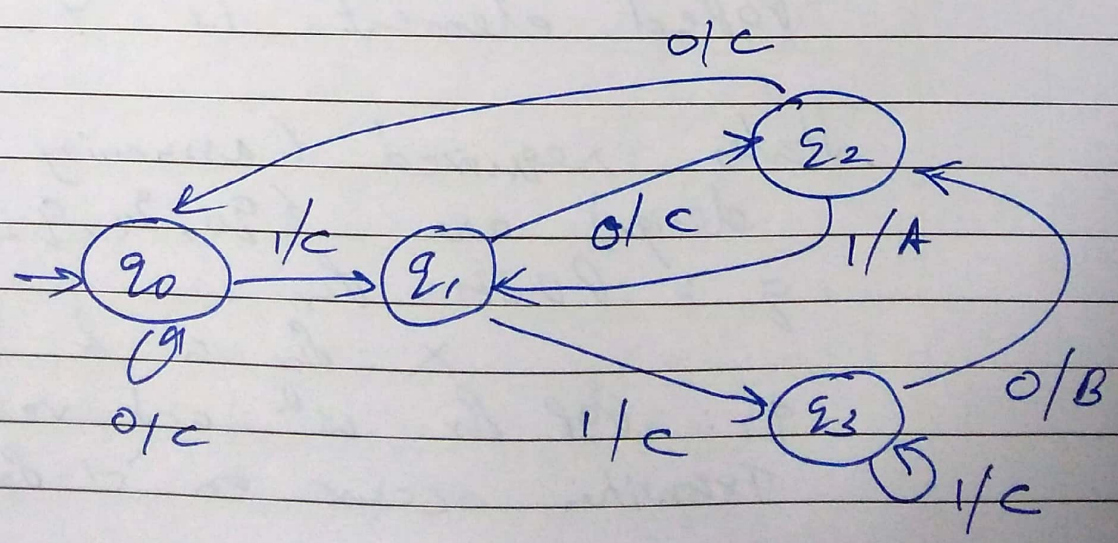


Minimization of above Mealy m/c gives

$\delta' : Q' \times \Sigma \to Q'$ $\qquad$ $\lambda' : Q' \times \Sigma \to \Delta$

Q' \ Σ	0	1
q_0	q_0	q_1
q_1	q_2	q_3
q_2	q_0	q_1
q_3	q_2	q_3

Q' \ Σ	0	1
q_0	c	c
q_1	c	c
q_2	c	A
q_3	B	c



Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

5a Design a PDA for the following language:

$L(M) = \{ wcw^R \mid w \in \{a,b\}^* \}$ where w^R is reverse of w and c is a constant.

Ans To match w^R with w, we push w in stack. as.

For each 'a' in string before 'c', push X. in stack
For each 'b' in string before 'c', push Y in stack.

After c is encountered. for each symbol in w^R, we pop and verify. that

For each input 'a', after c, the poped element is X
For each input 'b', after c, the poped element is Y.

States required (assuming final state design) are $\{q_0, q_1, q_2\}$
q_0 : PUSH for w
X for a & Y for b
q_1 : Pop for w^R and verify
Transition occurs on 'c' from q_0 to q_1

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

Design of final state PDA

$M = (Q, \Sigma, \tau, \delta, q_0, Z_0, F)$

where

Q : finite set of states are $\{q_0, q_1, q_2\}$

Σ : finite set of input symbols are $\{a, b, c\}$

τ : finite set of stack symbols are $\{X, Y, Z_0\}$

q_0 : Initial state

Z_0 : Bottom of stack symbol.

F : Final state is $\{q_2\}$

δ maps $Q \times \Sigma \cup \{\epsilon\} \times \tau \rightarrow Q \times \tau^*$

$\delta(q_0, a, Z_0) = (q_0, X Z_0)$

$\delta(q_0, b, Z_0) = (q_0, Y Z_0)$

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

$\delta(q_0, a, Y) = (q_0, X Y)$

$\delta(q_0, b, X) = (q_0, Y X)$

$\delta(q_0, b, Y) = (q_0, Y Y)$

$\delta(q_0, c, Z_0) = (q_1, Z_0)$

$\delta(q_0, c, X) = (q_1, X)$

$\delta(q_0, c, Y) = (q_1, Y)$

$\delta(q_1, a, X) = (q_1, \epsilon)$

$\delta(q_1, b, Y) = (q_1, \epsilon)$

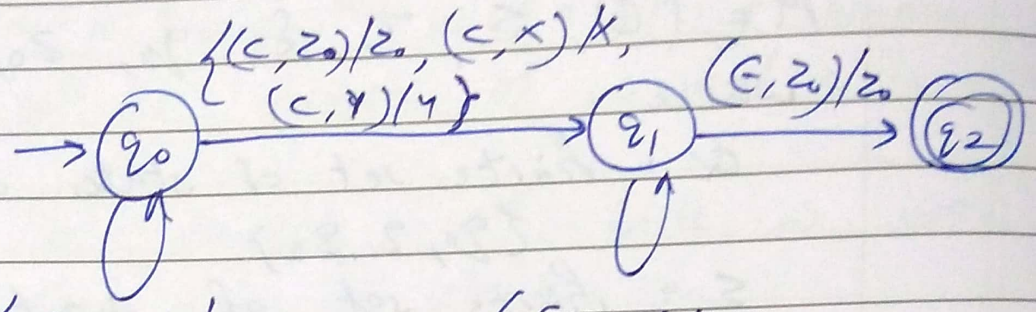
$\delta(q_1, \epsilon, Z_0) = (q_2, Z_0)$

Marks of no.		Examiner	
		Moderator	
		Re-Assessor	

Question No.

START WRITING HERE

Graphical representation



{(a, Z0)/XZ0,
(b, Z0)/YZ0,
(a, X)/XX,
(a, Y)/XY,
(b, X)/YX,
(b, Y)/YY}

{(a, X)/ε,
(b, Y)/ε}

Final PDA

5b) Convert the following grammar to the Chomsky Normal Form (CNF)

$S \to 0A0 \mid 1B1 \mid BB$

$A \to C$

$B \to S \mid A$

$C \to S \mid \epsilon$

Ans Every CFL without ε can be represented by a grammar with all productions of type $A \to BC$ and $A \to a$

where A, B, C are variables and a is a terminal.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

Consider grammar given

$S \to 0A0 \mid 1B1 \mid BB$
$A \to C$
$B \to S \mid A$
$C \to S \mid \epsilon$

1) Removal of ϵ production from grammar.
Remove $C \to \epsilon$ gives.

$S \to 0A0 \mid 1B1 \mid BB$
$A \to \epsilon \mid C$
$B \to S \mid A$
$C \to S$

Removal of $A \to \epsilon$ gives

$S \to \cancel{0A0}\ 0A0 \mid 1B1 \mid BB \mid 00$
$B \to S \mid \epsilon \mid \cancel{A} \mid A$
$C \to S$
$A \to C$

{A is eliminated as no prod.}

Removal of $B \to \epsilon$

$S \to 1B1 \mid BB \mid 00 \mid 11 \mid \epsilon \mid B \mid 0A0$
$B \to S \mid A$
$C \to S$
$A \to C$

~~C is unreachable and hence useless~~

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

$S \to 1B1 \mid BB \mid 00 \mid 11 \mid \epsilon$

$B \to S$

Removal of Useless variable 'B'

$\because$ $B \to S$ is only prod. of B.

$S \to 1S1 \mid SS \mid 00 \mid 11 \mid \epsilon$

~~Removal of $S \to \epsilon$~~

Removal of $A \to C$ and $C \to S$

Replacing A & C by S

$S \to 1B1 \mid BB \mid 00 \mid 11 \mid \epsilon \mid B \mid 0S0$

$B \to S \mid SX$ ie $B \to S$

Removal of $B \to S$

$S \to 1S1 \mid SS \mid 00 \mid 11 \mid \epsilon \mid S \mid 0S0$

Removal of $S \to \epsilon$

$S \to 1S1 \mid SS \mid 00 \mid 11 \mid \cancel{\epsilon} \mid \cancel{S} \mid 0S0$ (NOTE $S \to S$ is Useless Production)

Final simplified grammar is

$S \to 1S1 \mid 0S0 \mid SS \mid \cancel{S} \mid 00 \mid 11$

Examiner	
Moderator	
Re-Assessor	

START WRITING HERE

Converting in CNF form

$$\boxed{\begin{array}{l} S \to XSX \mid YSY \mid SS \mid XX \mid YY \\ X \to 1 \\ Y \to 0 \end{array}}$$

Reducing $S \to XSX \mid YSY$ into CNF we get

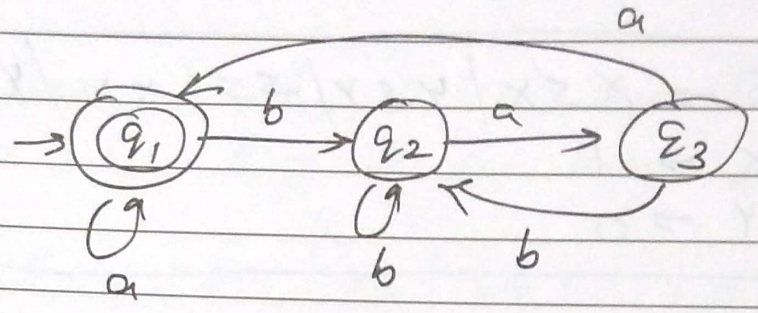
$$\boxed{\begin{array}{l} S \to XD_1 \mid YD_2 \mid SS \mid XX \mid YY \\ X \to 1 \\ Y \to 0 \\ D_1 \to SX \\ D_2 \to SY \end{array}}$$

Final CNF.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

3 a) Obtain a regular expression for the FA shown below:



Ans To convert the FA into RE.

Step 1 : Convert FA to left linear grammar form.

$$q_1 = q_1 a + q_3 a + \epsilon$$

$$q_2 = q_1 b + q_2 b + q_3 b$$

$$q_3 = q_2 a$$

Step 2

by Arden's Theorem.

$$R = RP + Q$$

is equivalent to $R = QP^*$

Applying Substitution and Arden's Theorem on the left linear grammar - above we get.

arks of on no.		Examiner	
		Moderator	
		Re-Assessor	

or	Question No.	START WRITING HERE

Substituting $q_3 = q_2 a$ in q_1 & q_2

$q_1 = q_1 a + q_2 aa + \epsilon$

$q_2 = q_1 b + q_2 b + q_2 ab$

Applying Arden's Theorem on q_2

$q_2 = q_2 (b + ab) + q_1 b$

by AT we get

$q_2 = q_1 b (b + ab)^*$

Substituting q_2 in q_1 we get

$q_1 = q_1 a + (q_1 b (b+ab)^*) aa + \epsilon$

Applying Arden's Theorem on q_1

$q_1 = q_1 (a + b(b+ab)^* aa) + \epsilon$

by AT we get

$q_1 = \epsilon (a + b(b+ab)^* aa)^*$

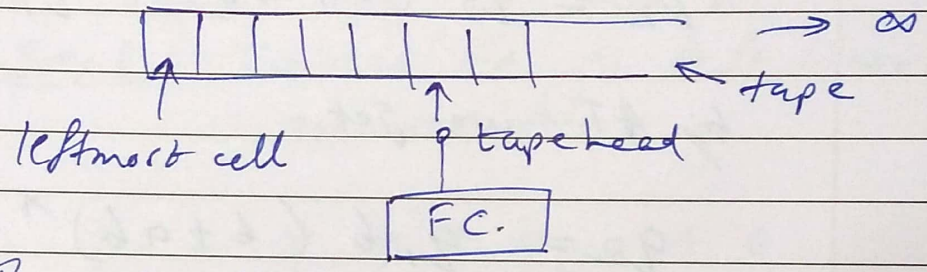
Final RE $\underline{(a + b(b+ab)^* aa)^*}$

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

3b) Explain the types of Turing machine in detail.

Ans The Basic model of Turing machine comprises of
1) A finite control
2) A read/write head
and 3) A single dimensional tape.



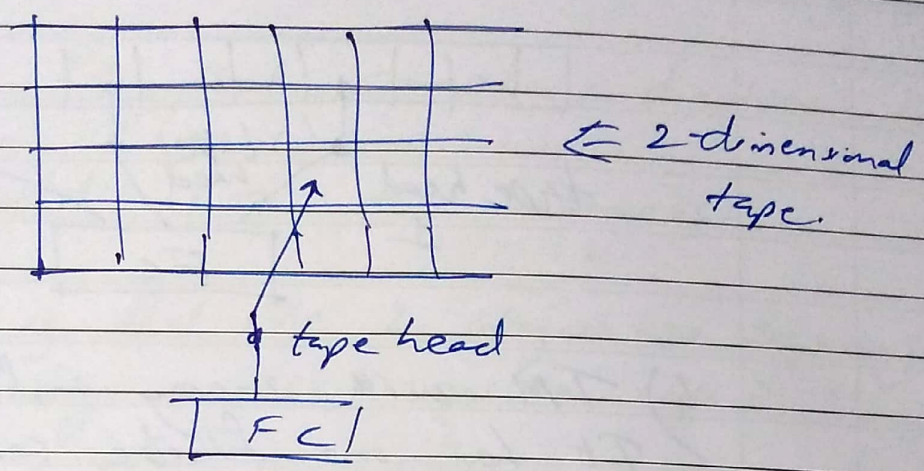
The head can read/write 1 cell of the tape and move 1 cell to right or left in a single transition.

Different variants/types of Turing m/cs are equivalent to basic model in power. They are

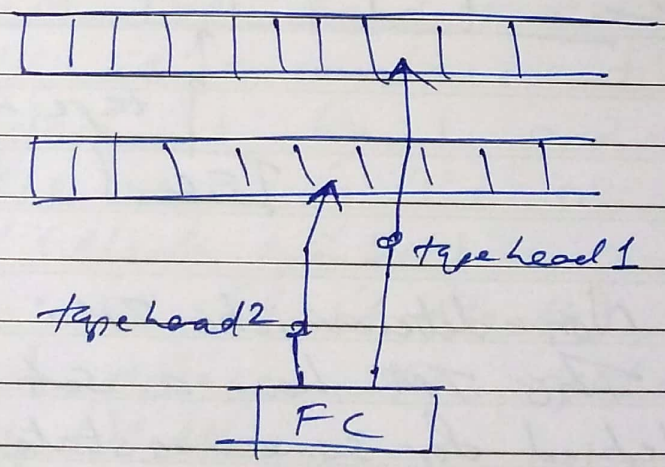
1) TM with two-dimensional tapes:-
It has finite control, a single read/write head and a 2-dimensional tape.
TM in a single transition can move head Left / Right / Up / Down

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE



2) TM with multiple tapes:-
It has one finite control, and more than one tapes. Each tape has a seperate read/write head and can be moved independently left/right

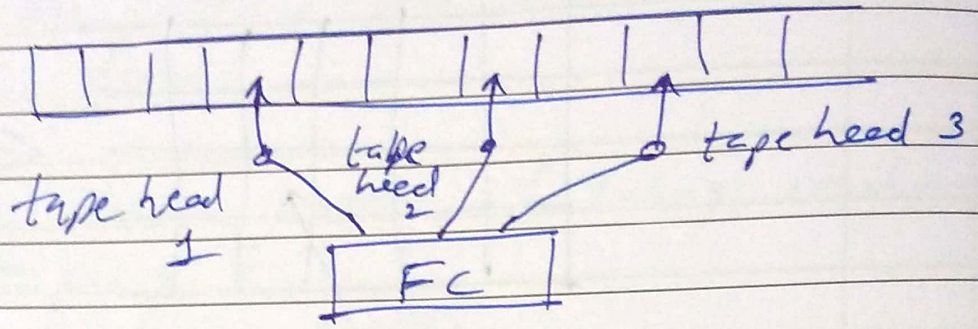


3) TM with multiple heads:
It has one finite control, one tape with more than one read/write heads. These heads can be moved independently left/right but on the same tape

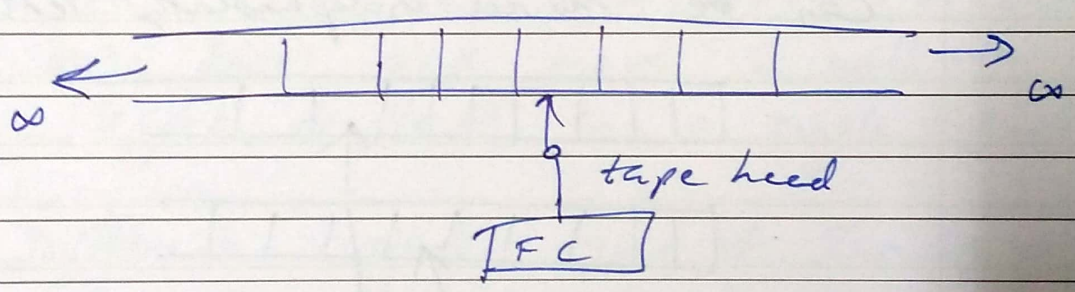
		Examiner	
of		Moderator	
		Re-Assessor	

Question No.

START WRITING HERE



4) TM with 2-way infinite tape :-
It has one finite control, one tape head and one single dimensional tape just like basic model. The Key difference is the tape has no leftmost cell and extends infinite on both sides.



5) Non-deterministic TM :
This TM has a set of actions defined for same ~~sa~~ state and same input. It has the ability to perform any action from a given set of actions rather than perform a definite predetermined (deterministic) action.

$$\delta(p,a) = \{(q,b,R), (r,c,L), \ldots\}$$

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

4)

a) Design a TM that computes a function ~~f(m+n)~~ $f(m,n) = m+n$.
i.e. addition of two integers.

Ans Assuming that the 2 integers are represented in Unary number system with $\Sigma = \{1\}$

i/p ⇒ $\overbrace{1111}^{4} + \overbrace{111}^{3}$

o/p (Required) ⇒ $\underbrace{1111111}_{7}$

Steps to perform addition.

1) Move type head to right for all '1's. till it encounters '+'.
2) Replace '+' by 1 and move right
3) Skip remaining '1's till it encounters 'B' (Blank symbol).
4) On 'B' (Right Blank), move left and replace the rightmost '1' by 'B'.
5) Stop./Halt.

States required $Q = \{q_0, q_1, q_2\}$

q_0 : Keeps '1's as ~~its~~ is and replaces '+' by '1'. till Blank 'B'.

q_1 : ~~Moves~~ Replaces last '1' by 'B'.

q_2 : Halt state.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

7 - tuple representation

$M = (Q, \Sigma, \tau, \delta, q_0, B, F)$

$Q = \{q_0, q_1, q_2\}$: As explained earlier

$\Sigma = \{1, +\}$: i/p symbols.

$\tau = \{1, +, B\}$: tape symbols

q_0 = Initial state & function as illustrated

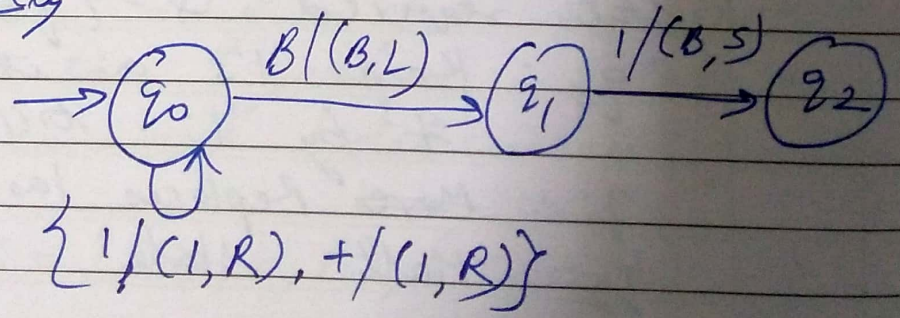
B = Blank symbol

$F = \phi$: No final state since it is TM as generator.

$\delta : Q \times \tau \rightarrow Q \times \tau \times \{L, R, S\}$

$Q \backslash \tau$	1	+	B
q_0	$(q_0, 1, R)$	$(q_0, 1, R)$	(q_1, B, L)
q_1	(q_2, B, S)	—	—
q_2	—	—	—

Graphically



Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Question No.	START WRITING HERE

Simulation for 3 + 2 with Ans as 5.

$$\overbrace{|||}^{3}+\overbrace{||}^{2}$$

$q_0 |||+||B$

$\vdash |q_0||+||B$

$\vdash ||q_0|+||B$

$\vdash |||q_0+||B$

$\vdash ||||q_0||B$

$\vdash |||||q_0|B$

$\vdash ||||||q_0B$

$\vdash |||||q_1|B$

$\vdash \underbrace{|||||}_{5}q_2BB$

Halt state

4b) State and explain Pumping Lemma for CFL. Find out whether the language $L = \{x^n y^n z^n \mid n \geq 1\}$ is context free or not.

Ans Pumping lemma for CFL states that for any CFL 'L', it is possible to find two substrings that can be 'pumped' any number of times and still be in the same language.

For any language L, we break its string into five parts and pump second and fourth substring.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

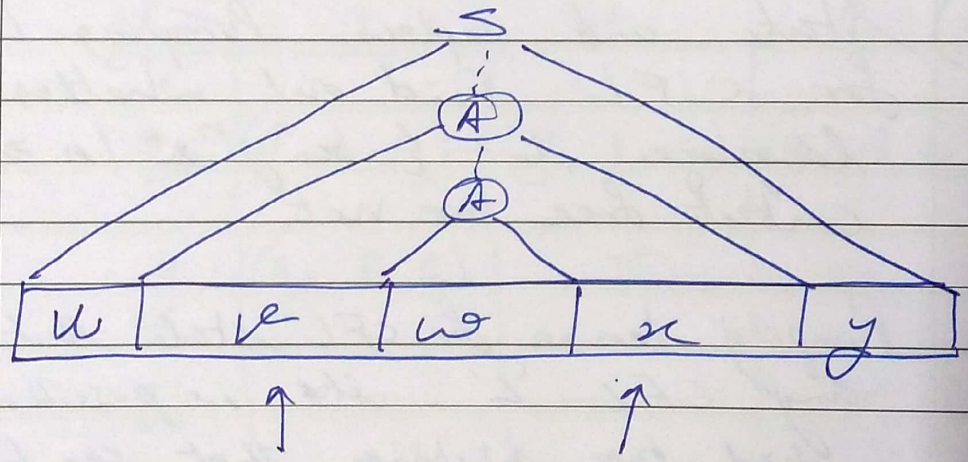
Pumping lemma for CFL is a tool to prove that a language is not CFL. Because, if one string does not satisfy its condition, then the language is not CFL.

Thus; if L is a CFL, there exists an integer 'n' such that ~~both~~ for all ~~z~~ $\in L$ with $|z| \geq n$, there exists $u, v, w, x, y \in \Sigma^*$, such that ~~z~~ $z = uvwxy$, and

i) $|vwx| \leq n$

ii) $|vx| \geq 1$

and iii) $\forall i \geq 0$, $uv^iwx^iy \in L$



Can be 'pumped' any number of times

$L = \{x^n y^n z^n \mid n \geq 1\}$ is not a CFL.

Proof:

Let K be the constant of PL for CFL

Let $z = x^K y^K z^K$

s of no.		Examiner
		Moderator
		Re-Assessor

Question No.

START WRITING HERE

$z \in L$ for $n = K$

$|z| = 3K \geq K$

$\therefore$ z must satisfy PL for CFL if 'L' is CFL

$z = uvwxy$

Condition ① $|vwx| \leq K$

hence vwx cannot contain all 3 symbols x, y and z. Since distance between last x and first z is greater ~~than~~ than 'K'.

Condition ② $|vx| \geq 1$

Hence vx contain atleast 1 symbol.

Condition ③ $\forall i \geq 0$ $uv^iwx^iy \in L$

Let $i = 0$ uwy must $\in L$

But uwy has atleast 1 symbol K times (as per condition 1) and atleast one symbol less than K times (as per condition 2).

Hence $uwy \notin L$

This contradicts PL for CFL.

Hence $\{x^n y^n z^n \mid n \geq 1\}$ is not a CFL.

ks of no.		Examiner	
		Moderator	
		Re-Assessor	

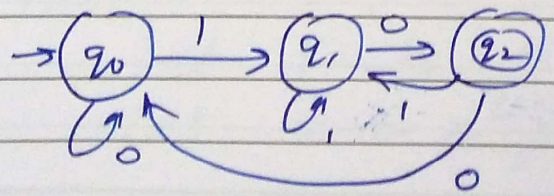
Question No. 1a)

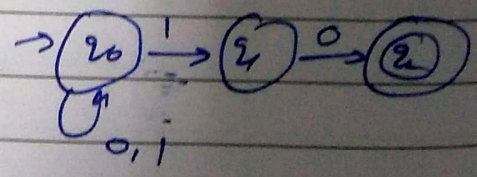
START WRITING HERE

Differentiate DFA and NFA.

Ans

DFA	NFA
1) Deterministic Finite Automata has exactly one transition from each state on an input	Non-Deterministic Finite Automata can have 0 to any no of transitions from a state on an input.
2) 5-tuple representation of DFA is $M = (Q, \Sigma, \delta, q_0, F)$ where δ maps $Q \times \Sigma$ to Q	5-tuple representation of NFA is $M = (Q, \Sigma, \delta, q_0, F)$ where δ maps $Q \times \Sigma \cup \{\epsilon\}$ to 2^Q
3) It cannot have ϵ-transition	ϵ-transitions are allowed in NFA.
4) A string is accepted by DFA if path from start state reaches a final state	A string is accepted by NFA if one amongst the multiple path ends on a final state
5) DFA can be complemented as $M = (Q, \Sigma, \delta, q_0, Q-F)$ to recognize $\overline{L(M)}$	NFA cannot be complemented without converting to equivalent DFA.
6) Example: DFA for strings ending with "10"	Example: NFA for strings over $\{0,1\}$ ending with 10





Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

16 Design a DFA to accept strings of 0's and 1's ending with the string 100.

Ans DFA is represented by 5-tuple representation

$M = (Q, \Sigma, \delta, q_0, F)$

where Q is finite set of states

Σ ~~if~~ is finite set of input symbols

δ maps $Q \times \Sigma$ to Q

q_0 is initial state

F is set of final states

for strings ending with '100'

$Q = \{q_0, q_1, q_2, q_3\}$

q_0 : Initial state

q_1 : Strings ending with 1

q_2 : —"— 10

q_3 : —"— 100 (Final state)

$\Sigma = \{0, 1\}$ and $F = \{q_3\}$

$\delta : Q \times \Sigma$ to Q

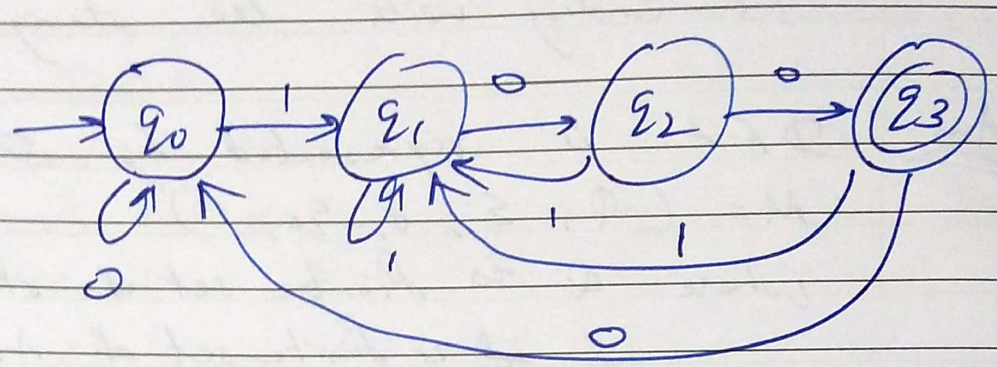
$Q \backslash \Sigma$	0	1
q_0	q_0	q_1
q_1	q_2	q_1
q_2	q_3	q_1
q_3	q_0	q_1

		Examiner	
s of no.		Moderator	
		Re-Assessor	

Question No.

START WRITING HERE

Graphical representation.



Simulation for "10100"

$\delta^*(q_0, 10100)$ $\quad \because \delta(q_0, 1) = q_1$

$= \delta^*(q_1, 0100)$ $\quad \because \delta(q_1, 0) = q_2$

$= \delta^*(q_2, 100)$ $\quad \because \delta(q_2, 1) = q_1$

$= \delta^*(q_1, 00)$ $\quad \because \delta(q_1, 0) = q_2$

$= \delta^*(q_2, 0)$ $\quad \because \delta(q_2, 0) = q_3$

$= q_3$

$\because q_3 \in F \quad \therefore 10100 \in L(M)$

ie Accepted

Simulation for "00"

$\delta^*(q_0, 00)$ $\quad \because \delta(q_0, 0) = q_0$

$= \delta^*(q_0, 0)$ $\quad \because \delta(q_0, 0) = q_0$

$= q_0$

$\because q_0 \notin F \quad \therefore 00 \notin L(M)$

ie Rejected

Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

START WRITING HERE

Question No. 1c

Explain the applications of Regular Expression

Ans

Regular Expression is a mathematical form of Regular Grammer that can represent any Regular Language which can be accepted / regonized by a DFA. RE is equivalent to DFA.

By definition

ϕ, ϵ and any $a \in \Sigma$ are Regular Expressions. If r and s are 2 REs then $r.s$ (Concatenation), $r+s$ (Union), r^* (Kleen closure) and r^+ (Positive closure) are Regular Expressions.

This mathematical form can be converted to a DFA as follows.

RE → NFA with ϵ → NFA w/o ϵ → DFA → Minimized DFA.

Hence RE is useful in representing any Regular language for which application needs a DFA to recognize. Such applications include.

1) Text processing tasks.
2) String Processing
3) Data Validation
4) Data Scraping.
5) Scanning in compilers.
6) Search engines.

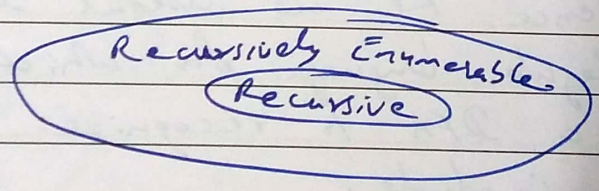
Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

1d What are Recursive and Recursively Enumerable Languages?

Ans Recursively Enumerable Languages are set of Languages that can be recognized by a Turing Machine. Hence given a RE language L_1, if any string $x \in L_1$ then on input x the TM for L_1 will halt and accept the input x. But consider a string $y \notin L_1$, then on input y the TM may or may not halt. ie TM for L_1 may loop forever.

Recursive Languages are a subset of Recursively Enumerable Languages (as shown below).



Given a Recursive language L_2, if any string $x \in L_2$ then on i/p x the TM for L_2 will halt and accept x. And on input $y \notin L_2$ the TM will halt and reject input y.
Hence TM for Recursive language will always halt.

	Examiner	
of	Moderator	
	Re-Assessor	

Question No.

START WRITING HERE

on every input.

Therefore if there exists a language L such that L belongs to RE but L is not Recursive then there exists atleast one input $y \notin L$ such that on input y the TM for L does not halt.

Due to this Recursive languages are closed under complement & Recursively Enumerable Language are not closed under complement.

If L is Recursive then $\bar{L}$ is Recursive

If L is RE but not Recursive then $\bar{L}$ is not RE.

~~Set~~ Set of Recursively Enumerable languages are called Turing recognizable languages whereas set of ~~Recursive~~ Recursive languages are called Turing Decidable languages.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

6a Write short notes ~~any~~ on (any two) :

a) Post Correspondance Problem :

The Post Correspondance Problem (PCP) is an ~~Undecida~~ Undecidable problem. It is stated as follows :-

Let Σ be an alphabet with at least 2 symbols. An instance of PCP is given by two sequences

$U = (u_1, u_2, \ldots, u_m)$ and
$V = (v_1, v_2, \ldots, v_m)$ of strings
where
$\forall i \; u_i, v_i \in \Sigma^*$.

Given that both U and V are sequences of same number of strings, the problem is to find ~~weath~~ whether there is a finite sequence $(i_1 \ldots i_p)$ with $i_j \in \{1..m\}$ for $j = 1..p$ so that

$$u_{i_1} u_{i_2} \ldots u_{i_p} = v_{i_1} v_{i_2} \ldots v_{i_p}$$

then $(i_1 \ldots i_p)$ is solution of PCP. and hence PCP has a solution.
The problem is to find whether PCP has a solution or not.

Examiner	
Moderator	
Re-Assessor	

Question No.

START WRITING HERE

For example: Consider the following sequences over $\Sigma = \{a, b\}$

	1	2	3	4	5	6
U	abab	aaabbb	aab	ba	ab	aa
V	ababaaa	bb	baab	baa	ba	a

then sequence 123 gives

$$U_{123} = \overbrace{abab}^{1}\,\overbrace{aaabbb}^{2}\,\overbrace{aab}^{3}$$

$$V_{123} = \underbrace{ababaaa}_{1}\,\underbrace{bb}_{2}\,\underbrace{baab}_{3}$$

$\therefore u_1 u_2 u_3 = v_1 v_2 v_3$

$\therefore$ 123 is a solution

Similarly 456 is also a solution of above PCP

Consider another example sequences over $\{a, b\}$

	1	2	3
U	ab	bab	bbaaa
V	a	ba	bab

In this case PCP has no solution.
If so, then a TM deciding solution will be in infinite loop.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

6b) Halting Problem:

~~Given~~ a Halting Problem is the problem of determining, from description of a computer program, and an input, whether the program will finish running (i.e halt) or continue to loop forever.

Alan Turing proved in 1936 that a general algorithm to solve the halting problem for all possible program-input pairs cannot exists. Hence Halting Problem was proven to be Undecidable over TM.

It is possible to run a program for an input and check whether it halts or not. But the answer to the question Given a program/algorithm will ever halt or not? is No. we cannot design a generalized algorithm which can appropriately say that given a program will ever halt or not.

To justify the answer, let us assume that given an algorithm / program 'A',

Examiner	
Moderator	
Re-Assessor	

Question No.

START WRITING HERE

we design a program 'B' to determine whether 'A' halts on all inputs or not.
In order to get this answer, 'B' calls 'A' for each possible input. ~~over Σ*~~.
For any input, ~~if~~ when 'B' calls 'A', if 'A' returns halt then 'B' calls 'A' for next input.
But, in case, 'A' runs in infinite loop over certain input passed by 'B'. then it is impossible for outer function 'B' to halt if its code (inner body 'A') is in infinite loop and therefore outer function 'B' also does not halt.

Hence halting problem is Undecidable.

The theoretical conclusion of not ~~sott~~ solvable or Unsolvable defines a class of applications which no programming invention can possibly perform perfectly.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

6c) Rice's Theorem:

Rice Theorem states that any non-trivial semantic property of a language which is recognized by a TM is undecidable. A property, P, is the language of all Turing Machine that satisfy that property.

Formal Definition:
If P is a non-trivial property, and the language holding the property L_P is recognized by TM M, then $L_P = \{ \langle M \rangle \mid L(M) \in P \}$ is undecidable.

→ Property of languages, P, is simply a set of languages. If any language belongs to P ($L \in P$) then it is said that L satisfies the property P.

→ A property is called to be trivial if either it is not satisfied by any recursively enumerable language; or it is satisfied by all recursively enumerable languages.

→ A non-trivial property is satisfied by some recursively enumerable language

Examiner	
Moderator	
Re-Assessor	

Question No.

START WRITING HERE

and are not satisfied by others.
Formally speaking, in a non-trivial property, where $L \in P$, both the following properties hold:

Property 1: There exists Turing Machines M_1 and M_2 that recognize the same language ie either
$(\langle M_1 \rangle, \langle M_2 \rangle \in L)$ or
$(\langle M_1 \rangle, \langle M_2 \rangle \notin L)$

Property 2: There exists Turing Machines M_1 & M_2, where M_1 recognizes the language while M_2 does not
ie $\langle M_1 \rangle \in L$ and $\langle M_2 \rangle \notin L$.

Rice's theorem generalizes the theorem that the halting problem is unsolvable.
Also note that the theorem holds only for properties of the partial function implemented by the program.
Rice's Theorem does not apply to properties of the program itself.