VIT Vidyalankar Institute of Technology

End Semester Examination (MU)

Evaluation of University Question Paper Solution Prepared by Faculty

Subject (Write in full) : Theory of Computer Science **(Regular / KT)**

Exam : May 20 19 **/ Nov. 20** — **Exam Date :** 27/05/2019 **Q. Paper Code :** 68476

Department : CMPN **Year : FE/SE/TE/BE/ME/MMS Semester :** V **Scheme : CBSGS/CBCS**

Handwritten Solution Prepared by : Ravindra S. Sangle

Name of the Subject Cluster : System Software

Name of the Cluster Mentor / Assessor : Prof. Pankaj Vanwari

1st Assessment :

Question No.	Marks Obtained						Total
1	(a) 05	(b) 05	(c) 05	(d) 04	(e)	(f)	19
2	(a) 10	(b) 10	(c)	(d)	(e)	(f)	20
3	(a) 10	(b) 09	(c)	(d)	(e)	(f)	19
4	(a) 09	(b) 09	(c)	(d)	(e)	(f)	18
5	(a) 10	(b) 10	(c)	(d)	(e)	(f)	20
6	(a) 09	(b) 09	(c) 09	(d)	(e)	(f)	27
				Total (Out of 130 marks)			123

Signature of Assessor / Cluster Mentor : Pankaj Vanwari - VNI

2nd Assessment :

Question No.	Marks Obtained						Total
1	(a)	(b)	(c)	(d)	(e)	(f)	
2	(a)	(b)	(c)	(d)	(e)	(f)	
3	(a)	(b)	(c)	(d)	(e)	(f)	
4	(a)	(b)	(c)	(d)	(e)	(f)	
5	(a)	(b)	(c)	(d)	(e)	(f)	
6	(a)	(b)	(c)	(d)	(e)	(f)	
				Total (Out of		marks)	

Signature of Assessor / Cluster Mentor : _____

Total Marks of Question no.	Examiner	
	Moderator	
	Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

(Q1) a) Differentiate DFA and NFA. (5m)

	DFA Deterministic FA	NFA Non-deterministic FA
(1)	In DFA, from each state on each i/p symbol there is exactly one transition	In NFA from each state on each i/p symbol there can be 0, 1 or more transition
(2)	In DFA, the transition function is defined as $\delta: Q \times \Sigma \rightarrow Q$	In NFA, the transition function is defined as $\delta: Q \times \Sigma \rightarrow 2^Q$
(3)	The implementation of DFA with the help of computer program is simple	The implementation of NFA with computer prog is difficult because of its non-deterministic nature.
(4)	Every DFA is always an NFA	An NFA may or may not be a DFA
(5)	It is easy to find whether WTL as transitions are deterministic	It is difficult to find whether WTL as there are several paths. Backtracking is req'd to explore several parallel paths.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

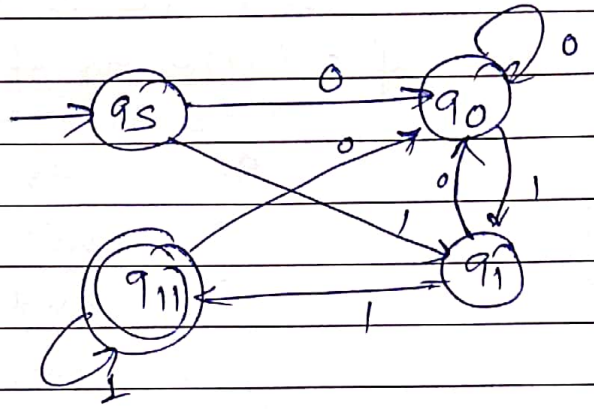
Space for Marks	Question No.	START WRITING HERE

Example.

i) DFA to accept a string ends with "11" over $\Sigma = \{0,1\}$

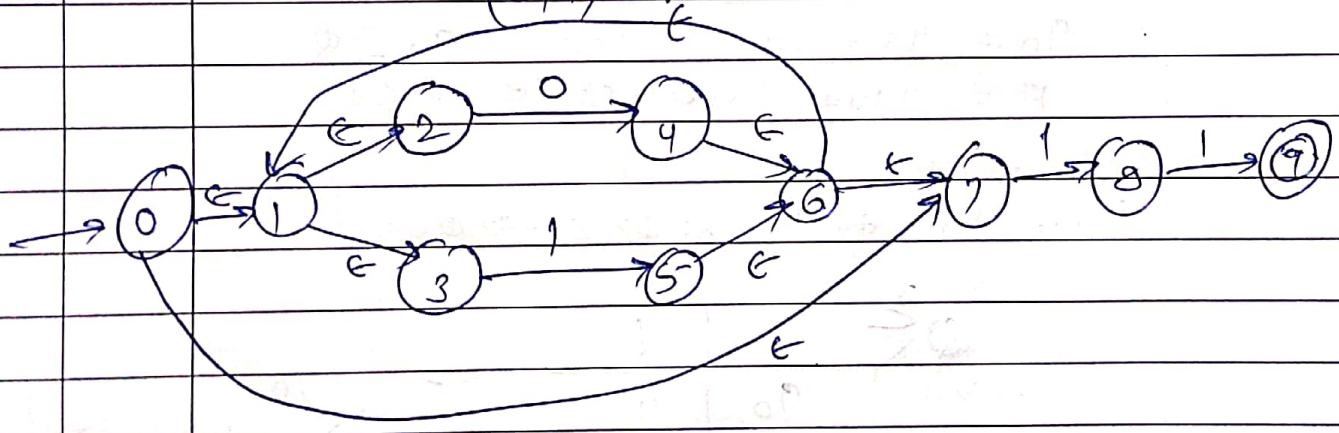
$\delta : Q \times \Sigma \rightarrow Q$

Q \ Σ	0	1
→ q_s	q_0	q_1
q_0	q_0	q_1
q_1	q_0	q_{11}
* q_{11}	q_0	q_{11}



ii) NFA to accept string ends with "11" $\Sigma = \{0,1\}$

$\Sigma = (0|1)^* 11$



Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE
	Q1) b)	Design a DFA to accept string of 0's and 1's ending with the string 100. (5m)
	Soln	DFA can be defined as a tuple.

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

Q = Finite no of states.

q_s = start state

q_0 = ends with 0

q_1 = ends with 1

q_{10} = ends with 10

q_{100} = ends with 100

Σ = finite set of i/p alphabet

$= \{0, 1\}$

δ = Transition function $Q \times \Sigma \to Q$

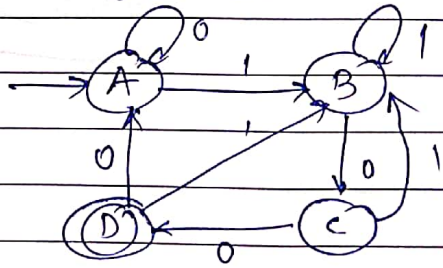
$q_0 = q_s$ = start state $q_0 \subseteq Q$

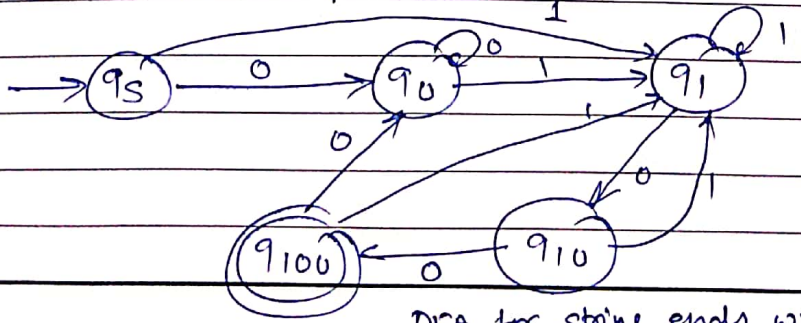
$F = q_{100}$ final state $F \subseteq Q$

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

Q / Σ	0	1
→ q_s	q_0	q_1
q_0	q_0	q_1
q_1	q_{10}	q_1
q_{10}	q_{100}	q_1
* q_{100}	q_0	q_1

DFA Min.





DFA for string ends with 100

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE
		Ex. $\delta(q_5, 01100)$ $\delta(q_5, 0)\,1100$ $\delta(q_0, 1)\,100$ $\delta(q_1, 1)\,00$ $\delta(q_1, 0)\,0$ $\delta(q_{10}, 0) \Rightarrow q_{100}$ ✓
	Q1) c)	Explain the applications of Regular Expressions. (5m)
	Sol^n	Regular expressions are useful in a wide variety of text processing tasks, and more generally string processing, where the data need not be textual. Common applications includes data validation, data scraping (especially web scraping), data wrangling, simple parsing, the production of syntax highlighting systems, and many other tasks While regular expressions would be useful on Internet search engines, processing them across the entire database could consume excessive computer resources depending on the complexity and design of the regular expression

Q1) d) What are Recursive and Recursively Enumerable languages? (5m)

Ans 1) A language is recursive if there exists a Turing machine that accepts every string in the language and rejects if it is not in the language.

2) For ex lets take turing machine M and string w:-
if string w is a member of the turing machine M, then M halts in its final state otherwise it rejects the computation.

3) A language is recursively enumerable if there exists a turing machine that accepts every string in the language and rejects if if is not in the language may be loop forever.

4) For ex. take TM 'M and string w:-

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE
		if string 'w' is in the language, then M halts in its final state. Otherwise it rejects the computation or may be run forever.
		5) Recursive languages are decidable by some TM ie there is a TM that can, given any i/p string correctly answer yes if the string is in the language, or no if it isn't.
		6) A language is recursive enumerable if there exist a TM that keeps outputting strings that belongs to the language, such that eventually every string in the language will be in the output.
		7) Recursively enumerable languages are only recognized ie there exist a TM that accepts when the string is in the language but it may loop forever if the string is not in the language.
		8) Recursively enumerable TM if it not accepts a string may halt in non-final state or loop forever which is not the case for recursive languages.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

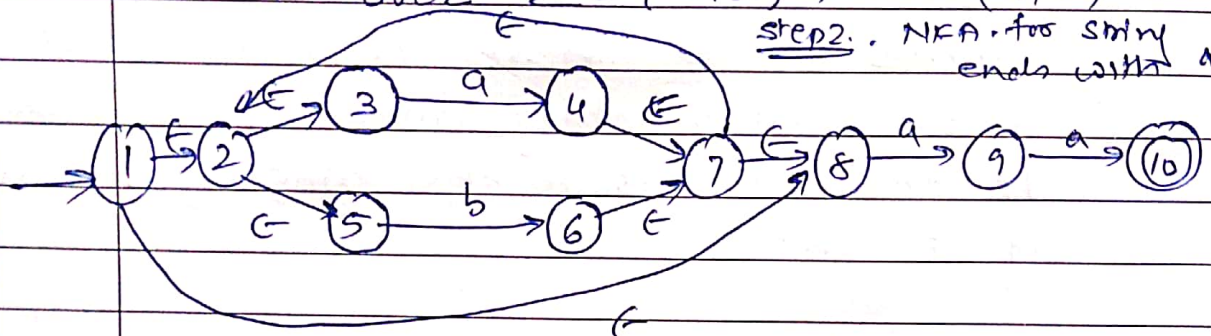
Space for Marks | Question No. | START WRITING HERE

Q2) a) Design NFA for recognizing the strings that ends in "aa" over $\Sigma = \{a, b\}$ and convert above NFA to DFA. (10m)

Soln

Step 1: R.E to accept a string ends with "aa" over $\Sigma = \{a, b\} \Rightarrow R = (a/b)^* aa$

Step 2: NFA for string ends with aa



NFA with ε-transitions

Step 3: NFA to DFA conversion

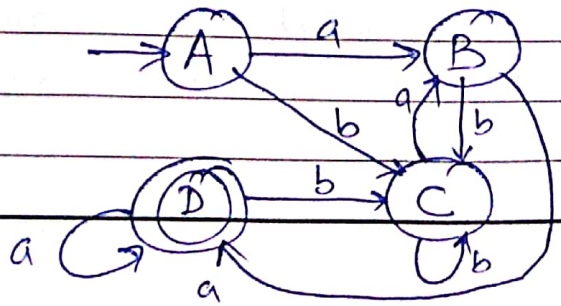
States(x)	Y = ε-closure(x)	δ(Y, a)	δ(Y, b)
→ {1} A	1, 2, 3, 5, 8	{4, 9}	{6}
{4, 9} B	4, 7, 8, 2, 3, 5, 9	{4, 9, 10}	{6}
{6} C	6, 7, 8, 2, 3, 5	{4, 9}	{6}
* {4, 9, 10} D	4, 7, 8, 2, 3, 5, 9, 10	{4, 9, 10}	{6}

Step 4:

DFA transition table $\delta: Q \times \Sigma \to Q$

Q \ Σ	a	b
→ A	B	C
B	D	C
C	B	C
* D	D	C

Step 5: DFA



Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks | Question No. | START WRITING HERE

Q2) b) Design moore m/c for following
If i/p ends in "101" then output A, if i/p ends in "110" o/p should be B, otherwise o/p should be C and convert it into mealey m/c (10m)

soln Mathematical model of moore m/c can be defined as.

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

Q = Finite no. of states
q_s = start state
q_0 = ends with 0
q_1 = ends with 1
q_{10} = Ends with 10
q_{101} = ends with 101
q_{11} = ends with 11
q_{110} = ends with 110

Σ = finite set of i/p alphabet = $\{0,1\}$
δ = Transition function $\delta: Q \times \Sigma \to Q$
Δ = output alphabet = $\{A, B, C\}$
λ = output mapping $\lambda(Q) = \Delta$
$q_0 = q_s$ = start state

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

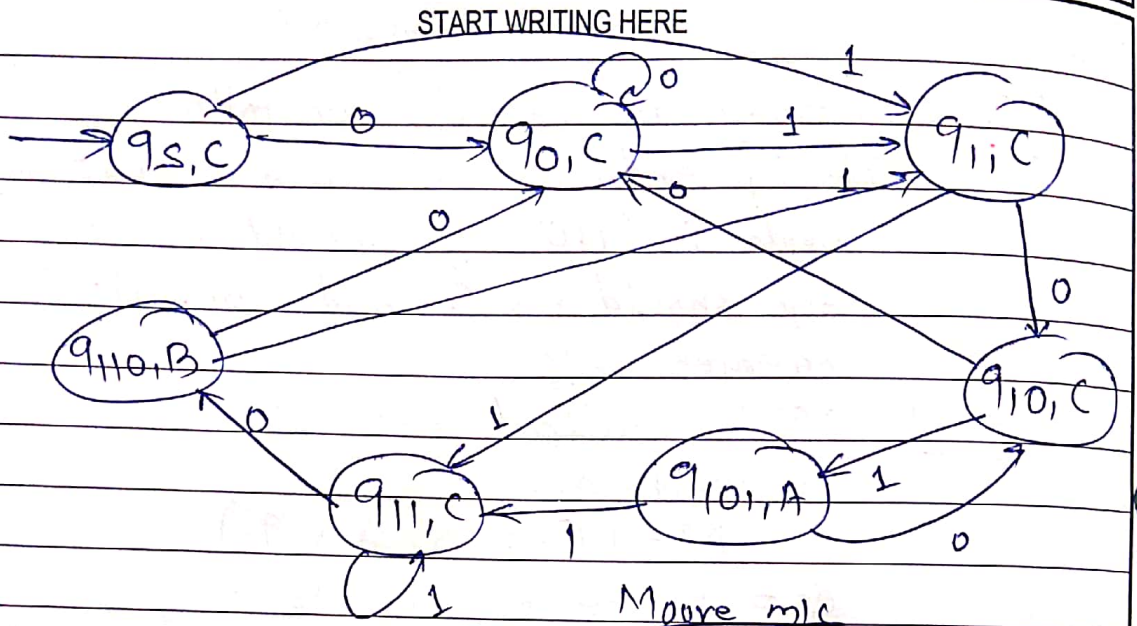
Q / Σ	0	1
C → q_s	q_0	q_1
C q_0	q_0	q_1
C q_1	q_{10}	q_{11}
C q_{10}	q_0	q_{101}
A q_{101}	q_{10}	q_{11}
C q_{11}	q_{110}	q_{11}
B q_{110}	q_0	q_1

output Mapping
$\lambda = \lambda(Q) = \Delta$
$\lambda(q_s) = C$
$\lambda(q_0) = C$
$\lambda(q_1) = C$
$\lambda(q_{10}) = C$
$\lambda(q_{101}) = A$
$\lambda(q_{11}) = C$
$\lambda(q_{110}) = B$

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.

START WRITING HERE



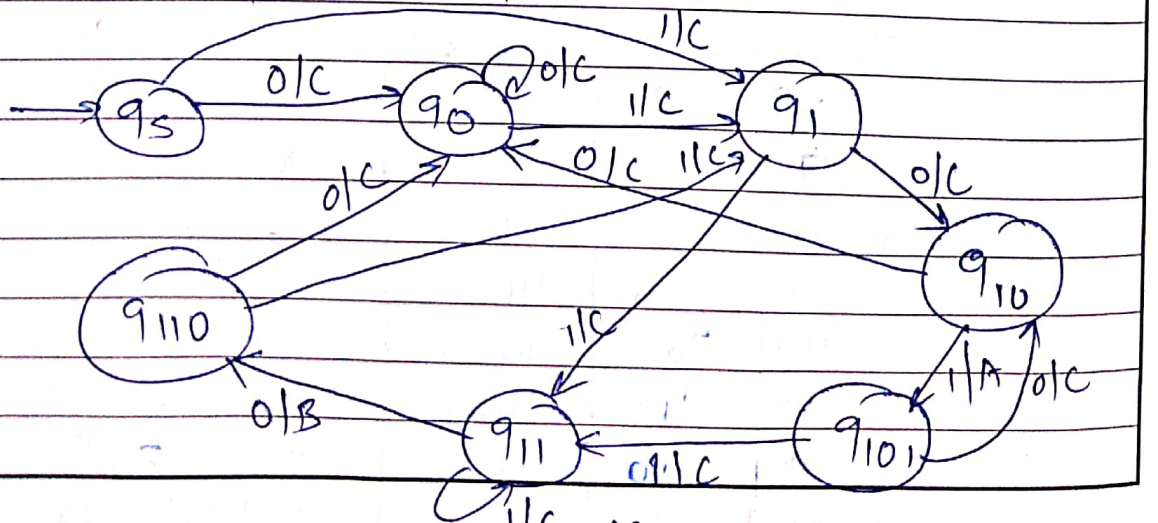
Moore m/c

Mealey m/c

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

λ = output mapping $\therefore \lambda : Q \times \Sigma \to \Delta$

Q \ Σ	0	1
→ q_s	q_0 C	q_1 C
q_0	q_0 C	q_1 C
q_1	q_{10} C	q_{11} C
q_{10}	q_0 C	q_{101} A
q_{101}	q_{10} C	q_{11} C
q_{11}	q_{110} B	q_{11} C
q_{110}	q_0 C	q_1 C

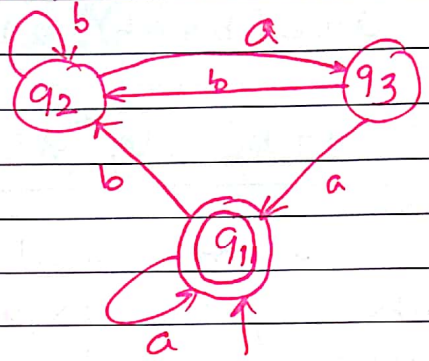


Mealey m/c

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

Q3) a) Obtain a regular expression for the FA shown below. (10m)



Soln The state equations of the states q_1, q_2 & q_3 are as follows –

$q_1 = q_1.a + q_3.a + \epsilon$ ——— (1)

$q_2 = q_1.b + q_2.b + q_3.b$ ——— (2)

$q_3 = q_2.a$ ——— (3)

Sub. for q_3 in the eqn for q_1 we get-

$q_1 = q_1.a + q_2.aa + \epsilon$ ——— (4)

Sub. for q_3 in eqn for q_2 we get-

$q_2 = q_1.b + q_2.b + (q_2.a)b$

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

$= q_1 b (b + ab)^*$ [By Arden's thm] ——— (5)

Now sub for q_2 in the eqn for q_1 ie eqn 4

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

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

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

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

By using Arden's theorem.

$$q_1 = \epsilon\left[a + b(b+ab)^* aa\right]^*$$

$$\therefore \boxed{q_1 = (a + b(b+ab)^* aa)^*}$$

As q_1 is the only final state of the DFA, the regular expression R for the language accepted by DFA is,

$$\boxed{R = (a + b(b+ab)^* aa)^*}$$

(Q3) b) Explain the types of Turing machine in detail. (10m)

Soln 1) Turing machine with two way infinite tape:-

A turing machine with two way infinite tape is denoted by $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ as in the original model.

As its name implies, the tape is infinite to the left as well as to the right. We imagine that there is infinity of blank cells to the left and the right of the current non blank portion of the tape. L is recognized by a turing machine

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

with a two-way infinite tape iff it is recognized by a TM with a one-way infinite tape.

2] TM with Semi-infinite tapes:-

Till now, the TM tape head was allowed to move either left or right from its initial position, it is only necessary that the TM's head be allowed to move within the position at and to the right of the initial head position. In TM with semi-infinite tape there are no cells to the left of the initial positions.

3] Multitape Tyring machine..

A multitape Tyring machine consist of finite control with k tape head and K-tapes; each tape is indefinite in both the direction.

On a single move, depending on the state of finite control and the symbol of each of the tape heads, the m/c can

- Change state
- print a new symbol on each of the cells scaned by its tape heads.
- Move each of its tape heads, indepe, one cell to the left or right, or keep it stationary.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

4] Multihead Turing Machine.

A K-head TM has fixed no. of heads. The heads are numbered 1 through K and a move of the TM depends on the state and on the symbol scanned by each head. In one move, the heads may each move indep L, R or S.

5] Nondeterministic TM.

A NDTM is a device with a finite control and a single one-way infinite tape. For a given state and tape symbol scanned by tape head, m/c has finite no. of choice for the next move as new state, tape symb. to print and direction of head motion

6] Multidimensional TM.

It is a device having finite control and the tape consist of k-dim array of cells infinite in all 2k directions for some fixed k. Depending on the state & symb scanned the device can change state, print new symb, move tape head in 2k directions.

7] Composite TM.

Two or more TM can be combined to solve a collection of simpler probs so that the o/p of one TM forms the i/p to the next TM and so on. This is called as composition.

The idea of composite TM gives rise to the concept of breaking the complicated job into no. of jobs implementing each separately and then combining them together to get the jobs req'd to be done

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

Q4) a) Design a tusing machine that computes a function $f(m,n) = m+n$ ie addition of two integers (10m)

Sol^n

Function for TM $\Rightarrow f(m,n) = m+n$

Assume that m and n are unary numbers.

The tusing m/c M is given by :-

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

Q = finite set of states = $\{q_0, q_1, q_2, q_3\}$

Σ = finite set of i/p alphabets = $\{0, \#\}$

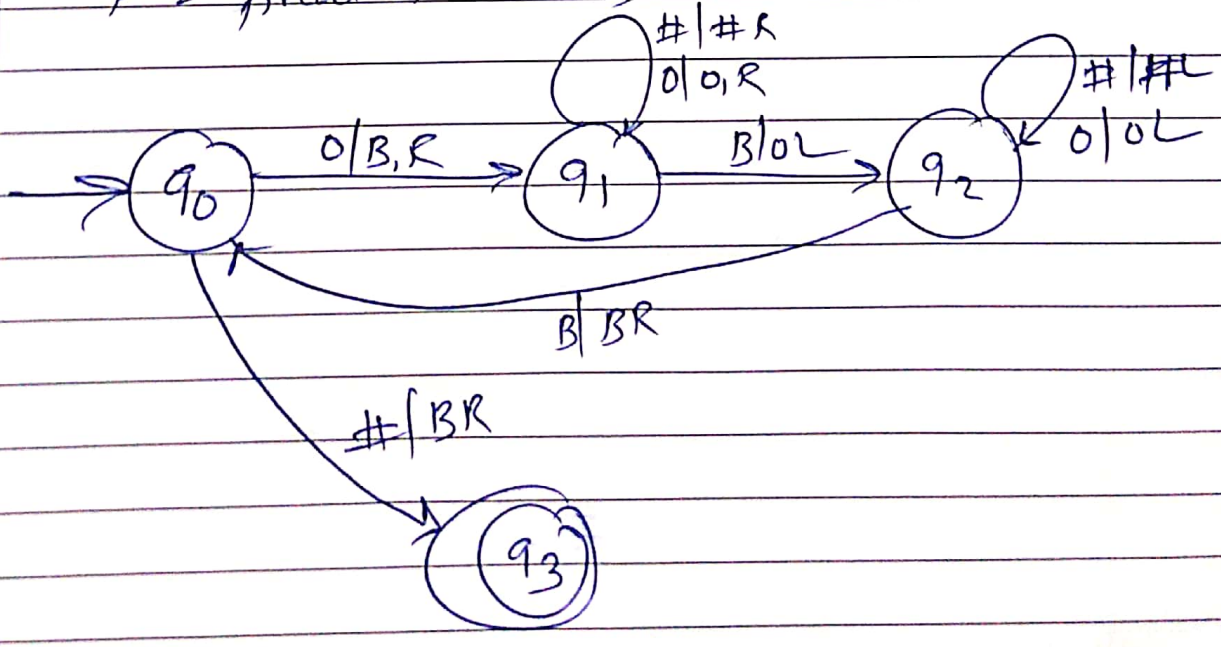
Γ = Tape alphabet = $\{0, \#, B\}$

δ = Transition functions

q_0 = start state

B = Blank symbol

F = final state = q_3



Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

Transition table

	0	#	B
→ q_0	q_1, BR	q_3, BR	—
q_1	q_1, 0R	q_1, #R	q_2, 0L
q_2	q_2, 0L	q_2, #L	q_0, BR
* q_3	q_3	q_3	q_3 ⇐ Stop condition

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE
	(Q4) b)	State and explain pumping lemma for context free languages. Find out whether the language $L = \{x^n y^{n^2} / n \geq 1\}$ is a context free or not (10m)
	Soln	Pumping lemma for CFL states that for any context free language 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.
		Pumping lemma, here also is used as a tool to prove that a language is not context free language. Because, if any one string does not satisfy its conditions, then the language is not CFL.
		Thus if L is in CFL, there exists an integer n, such that for all $\alpha \in L$ with $\|\alpha\| \geq n$, there exist $u, v, w, x, y \in \Sigma^*$ such that $\alpha = uvwxy$ and i) $\|vwx\| \leq n$ ii) $\|vx\| \geq 1$ iii) For all $i \geq 0$; $uv^i wx^i y \in L$

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

Let $L = \{x^n y^n z^n \mid n \geq 1\}$

Let L is context free. Then L must satisfy pumping lemma.

Take $w = x^n y^n z^n$

then break w as uvwxy such that

$|vwx| \leq n$ and $vx \neq \epsilon$

Hence vwx can not involves both x and z, since the last x and first z are atleast $(n+1)$ position apart.

There are two cases —

Case 1: vwx has no z. Then vx has only x and y. Then uwx, which would have to be in L, has n z's but fewer than n x's or y's

Case 2: vwx has no x's

Here contradiction occures,

Hence, L is not a context-free language.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

Q5) a) Design PDA for the following language $L\{M\} = \{W c W^R / W \in \{a,b\}^*\}$ where W^R is a reverse of W and c is constant. (10m)

5m

Mathematical model of DPDA can be defined as

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

where Q = finite set of states

Σ = finite set of i/p alphabets = $\{a, b, c\}$

δ = Transition function $\delta: Q \times \Sigma \to q\Gamma$

Γ = tape alphabet = $\{x, y\}$

q_0 = start state

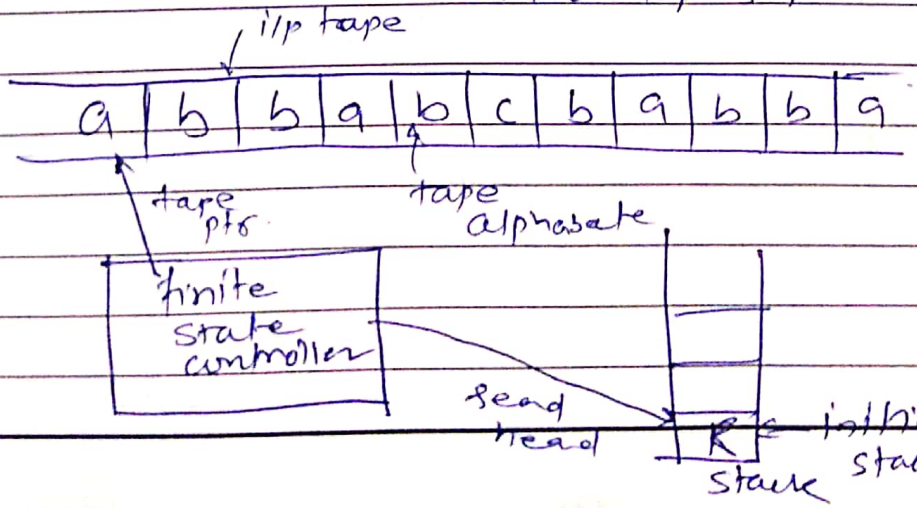
R = initial stack top [stack is empty]

F = final state

$L(M) = \{W c W^R / W \in (a,b)^*\}$

Logic - In W ⇒ for each 'a' push 'x'
for each 'b' push 'y'
for 'c' bypass the symbol

for W^R ⇒ for each 'a' pop x provided stack top is x
for each 'b' pop 'y' provided stack top is 'y'



Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks | Question No. | START WRITING HERE

Transition function

push $\delta(q_0, a, R) = (q_0, XR)$

$\delta(q_0, b, R) = (q_0, YR)$

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

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

push $\delta(q_0, b, X) = (q_0, YX)$

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

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

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

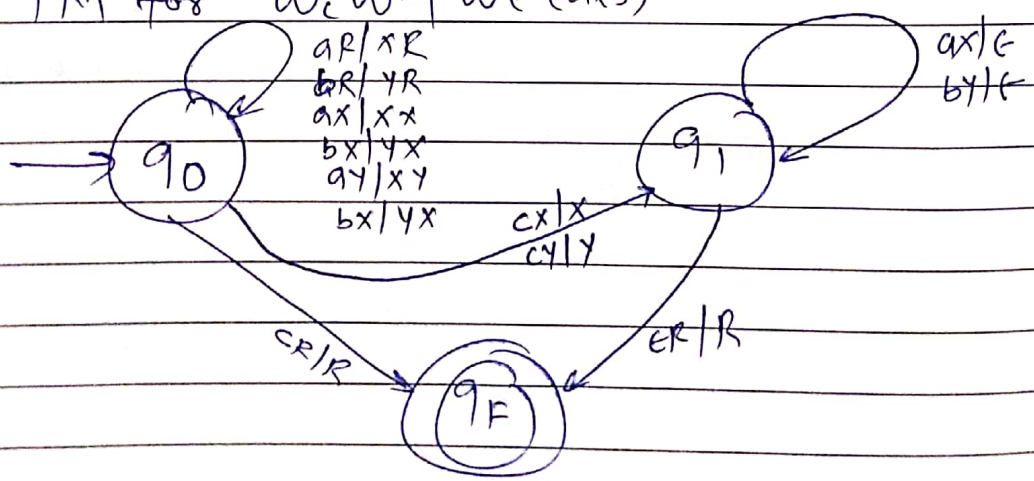
pop $\delta(q_1, a, X) = (q_1, \epsilon)$

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

$\delta(q_1, \epsilon, R) = q_F, R \Rightarrow$ Accept

$\delta(q_0, c, R) = q_F, R \Rightarrow$ Accept

TM for $W c W^R \mid W \in (a,b)^*$



ex. $\delta(q_0, abbcbba, R)$

↓ push X

$\delta(q_0, bbcbba, XR)$

↓ push Y

$\delta(q_0, bcbba, YXR)$

↓ push Y

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

$\delta(q_0, cbba, YYXR)$

⇓ nop & change state

$\delta(q_1, bba, YYXR)$

⇓ pop Y

$\delta(q_1, ba, YXR)$

⇓ pop Y

$\delta(q_1, a, XR)$

⇓ pop X

$\delta(q_1, \epsilon, R)$

Accept

Q5) b) Convert the following Grammar to the Chomsky normal form (CNF)

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

$A \rightarrow C$

$B \rightarrow S \mid A$

$C \rightarrow S \mid \epsilon$

Soln

Consider a CFG as $G = (V, T, P, S)$

V = Set of Variables = $\{S, A, B, C\}$

T = set of terminals = $\{0, 1, \epsilon\}$

P = Productions Rule

S = start variable

In the above CFG $C \rightarrow \epsilon$ is a null production and 'C' is nullable variable

As $A \rightarrow C \rightarrow \epsilon$

$B \rightarrow A \rightarrow C \rightarrow \epsilon$

$S \rightarrow BB \rightarrow \epsilon$

∴ A, B, C and S all are nullable variables

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

After elimination of nullable variables the resultant CFG becomes

$S \rightarrow 0A0 \mid 00 \mid 1B1 \mid 11 \mid BB \mid B$

$A \rightarrow C$

$B \rightarrow S \mid A$

$C \rightarrow S$

Here $A \rightarrow C$, $C \rightarrow S$, $B \rightarrow S$, $S \rightarrow B$ are unit productions.

So after elimination of unit prodn the resultant CFG is,

$S \rightarrow 0A0 \mid 00 \mid 1B1 \mid 11 \mid BB$

$\because A \rightarrow C \rightarrow S$ & $B \rightarrow S$

$\therefore$ Resultant CFG

$S \rightarrow 0S0 \mid 00 \mid 1S1 \mid 11 \mid SS$

CNF - [$A \rightarrow a$ or $A \rightarrow CD$]

$S \rightarrow C_0C_1$ ~~$S \rightarrow X_0C_1 \mid XX \mid CC_2 \mid 44 \mid SS$~~

$C_1 \rightarrow S$ ~~X_0~~

~~X_1~~ $\rightarrow 0$

$C \rightarrow 1$

$C_2 \rightarrow S4$

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE
	Q6) a)	Write a detailed note on (any two)
	a)	**Post corresponding problem.:**
	Soln	Let $A = w_1, w_2, \dots w_k$ and $B = x_1, x_2, \dots x_k$ be string over some alphabet Σ. Post correspondance problem (PCP) is to find the corresponclance sequenee of integers, $i_1, i_2, i_3 \dots i_m$ for $m \geq 1$ such that— $w_{i1}, w_{i2}, \dots w_{im} = x_{i1}, x_{i2}, \dots x_{im}$.
		The sequence $i_1, i_2, \dots i_m$ is considered to be a solution for the PCP instance. Each PCP instance is constituted by some set of values for A and B.
		Note that the entire class of PCP instances is unsolvable. If we consider the PCP as a generic class of all such instances, then it is unsolvable. Furthermore, there exist no generic algo that can find a solution for any such PCP instance; hence, it is also an undecidable problem. However, for a few val of A & B, it might have a soln.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

Consider the follo ex.

a) Post correspondance problem with solution

Let $\Sigma = \{0,1\}$ and let A and B be defined as table below

i	A	B	
	w_i	x_i	find the
1	0	000	correspondance
2	01000	01	sequence of ints
3	01	1	$i_1, i_2, \ldots i_m$ for $m \geq 1$
			such that —

$w_{i_1}, w_{i_2}, \ldots, w_{i_m} = x_{i_1}, x_{i_2} \ldots x_{i_m}$

Soln The given PCP instance has a solution for $m = 4$

$i_1 = 2$

$i_2 = 1$

$i_3 = 1$

$i_4 = 3$

Observe that —

$w_2 w_1 w_1 w_3 = (01000)(0)(0)(01) =$

$0\,1\,0\,0\,0\,0\,0\,0\,1$

$x_2 x_1 x_1 x_3 = (01)(000)(000)(1) =$

$0\,1\,0\,0\,0\,0\,0\,0\,1$

Hence $w_2 w_1 w_1 w_3 = x_2 x_1 x_1 x_3$

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

b) Post correspondance problem without soln

let $\Sigma = \{1, 0\}$ and let A and B be defined as table below

i	A	B
	w_i	x_i
1	ba	bab
2	abb	bb
3	bab	Abb

Find correspondance sequence of int. $i_1, i_2 \cdots i_m$ for $m \geq 1$ such that,

$w_{i_1}, w_{i_2} \cdots w_{i_m} = x_{i_1}, x_{i_2} \cdots x_{i_m}$

Soln This PCP instance is unsolvable since we can not find any correspondance sequence of integers as req^d.

Therefore every instance of ~~PCP~~ the given PCP might not have a solution. In other words the given PCP is undecidable that means given PCP does not have any algorithmic soln.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE
	(P6) b)	**Halting Problem.**

For a given initial configuration of a TM, two cases arise,

i) The machine starting at this configuration will halt after a finite number of steps

ii) The machine starting at this configuration never halts no matter how long it runs.

Given any TM, the problem of determining whether or not it ever halts is called the Halting problem.

To solve halting problem, we need some mechanism that consumes, any functional matrix, input data tape, and the initial configuration of TM, and determines whether or not the TM will ever halt. Any such mechanism essentially should be an algo (TM) that needs to simulate any given TM for checking whether or not it halts.

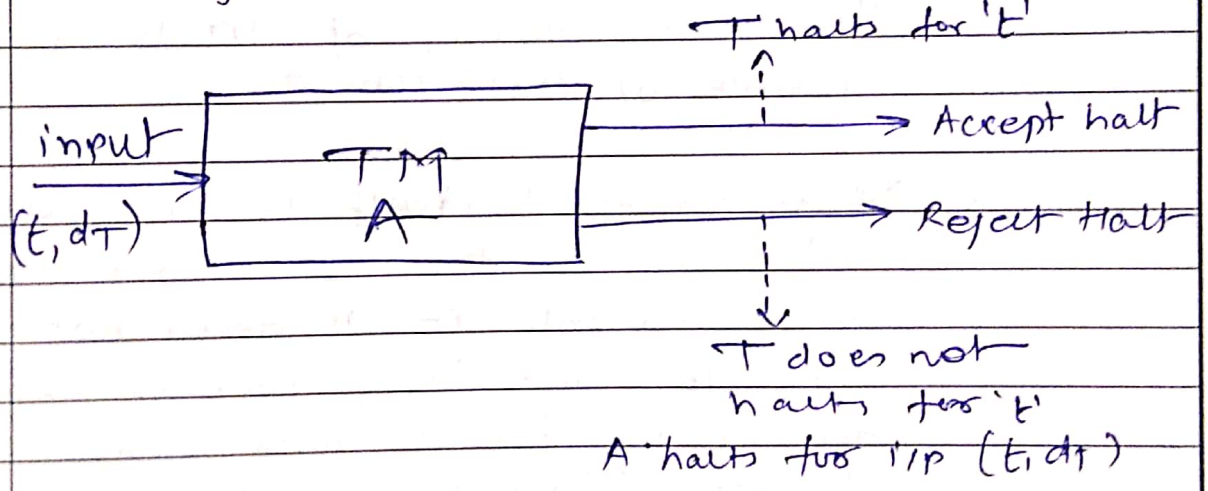
In reality one cannot solve the halting problem - one halting problem is unsolvable. This means that there exist no TM, which can determine whether or not a given TM will ever halts

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

Consider that the halting problem is unsolvable by contradiction.

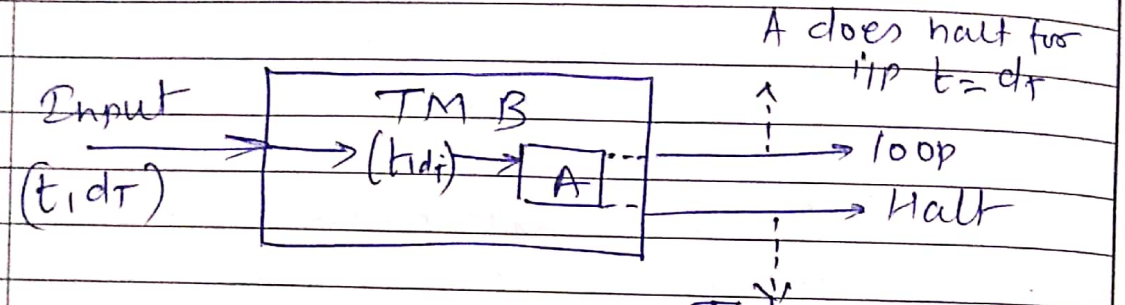
1) Assume that there exist a TM A, which decides whether or not any computation by any TM T will ever halt, given the description d_T ⟨FM⟩ of T, and the i/p tape 't' of T. Then for every i/p (t, d_T) to A, if T halts, then A reaches an "accept halt" state; else, A reaches a "reject halt" state as shown fig below



We now attempt to construct another TM B, which takes (t, d_T) as i/p, it functions as follows:-
first it copies the i/p and duplicates the same onto its tape. Then it takes this duplicated info tape as the i/p to A. Whenever A reaches the "accept halt" state, B loops forever, and whenever A reaches the "reject halt" state, B halts.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

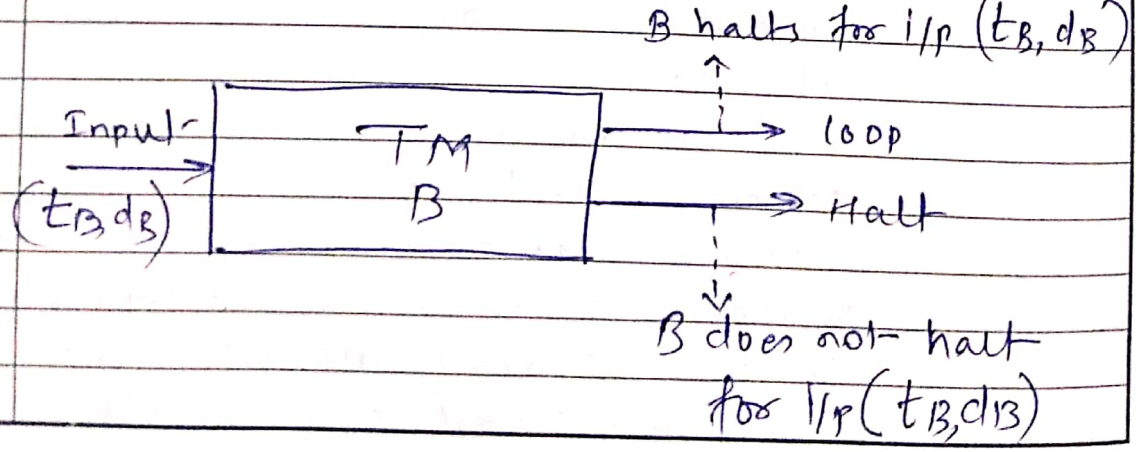
Space for Marks	Question No.	START WRITING HERE



Considering the original behaviour of A, we find that B acts as follows;

It loops if T halts for i/p t, and halts if T does not halt for i/p t. Thus the working of TM B is exactly opposite of that TM T.

Now since B itself is TM, let us set T = B. In this case, B halts for the i/p if and only if B does not halts for same i/p, and loops forever if B halts for the input. Fig follo shows working of ~~Thm~~ TM B, which takes itself as i/p.



Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

This is contradiction.

Hence we conclude that m/c A, which can decide whether or not any other TM will ever halt cannot exists.

Therefore, we conclude that the halting problem is unsolvable.

(Q6) c)

Rice's Theorem :-

A property of RE languages forms a set of RE languages. Thus if being a regular language is the property of the RE language, then it forms a set of all regular languages. A property of RE languages is said to be trivial if it is empty (ie not satisfied by any RE languages), or if it is satisfied by all RE languages. Otherwise it is said to be non-trivial.

Statement of Rice's theorem-

Every non-trivial property of an RE language is undecidable.

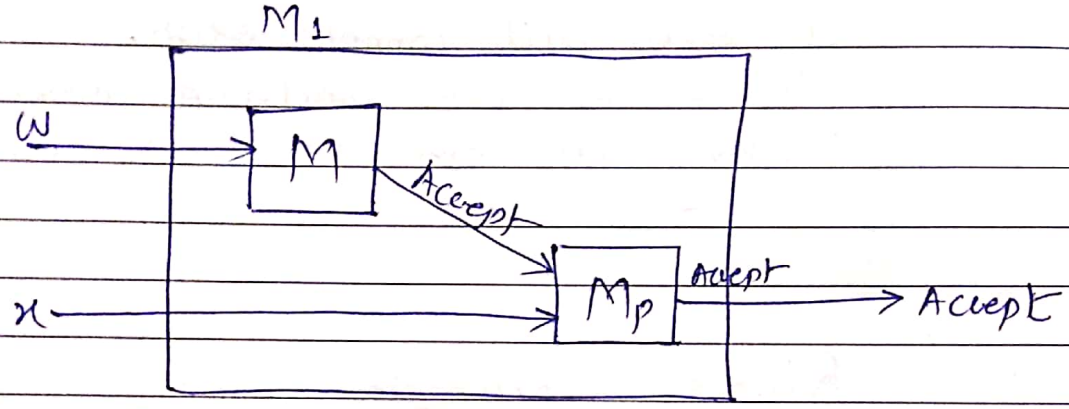
proof :-

To prove ~~st~~ Rice's theorem, we need to show that L_P is undecidable for a non-trivial property P. We shall proceed by reducing

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

Lu to Lp. As we know, if Lu is undecidable, so is Lp. To demonstrate this theorem we construct TM M1 as shown below



TM M1 that proves Lp is undecided.

The construction of M1 is such that if M does not accept w, then $L(M_1) = \phi$; if M accepts w, then $L(M_1) = L_p$.

On input (M,w) we create a TM M1 as follows:-

i) On input x, let the TM run on the string w until it accepts the string

ii) Next, run Mp on x. Accept it iff Mp does.

Note that M1 accept the same language as Mp if M accepts w; M1 accepts the empty language if M does not accept w.

Thus if M accepts w, the TM M1 has the property P; else it does not have it.

Total Marks of Question no.		Examiner	
		Moderator	
		Re-Assessor	

Space for Marks	Question No.	START WRITING HERE

This turns M_1 into M_P iff (M, w) is in L_u. Hence L_u is ~~de~~ reduced to L_P. This proves that the property P is undecidable just like L_u.

The essence of rice theorem is that any problem which requires determining the property of the language recognized by a given TM, is undecidable. The only exceptions are the trivial properties that are always either true or false.