Chapter 5
TURING MACHINE

Theory of computation

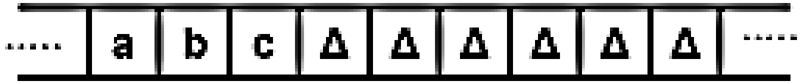
5.1 Turing Machine Introduction

- Turing machine was invented in 1936 by **Alan Turing**. It is an accepting device which accepts Recursive Enumerable Language generated by type 0 grammar.
- There are various features of the Turing machine:
- It has an external memory which remembers arbitrary long sequence of input.
- It has unlimited memory capability.
- The model has a facility by which the input at left or right on the tape can be read easily.
- The machine can produce a certain output based on its input. Sometimes it may be required that the same input has to be used to generate the output. So in this machine, the distinction between input and output has been removed. Thus a common set of alphabets can be used for the Turing machine.

Basic Model of Turing machine

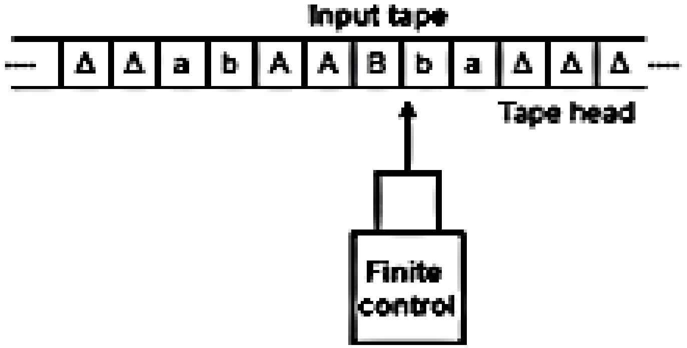
The turning machine can be modelled with the help of following representation.

1. The input tape is having an infinite number of cells, each cell containing one input symbol and thus the input string can be placed on tape. The empty tape is filled by blank characters.

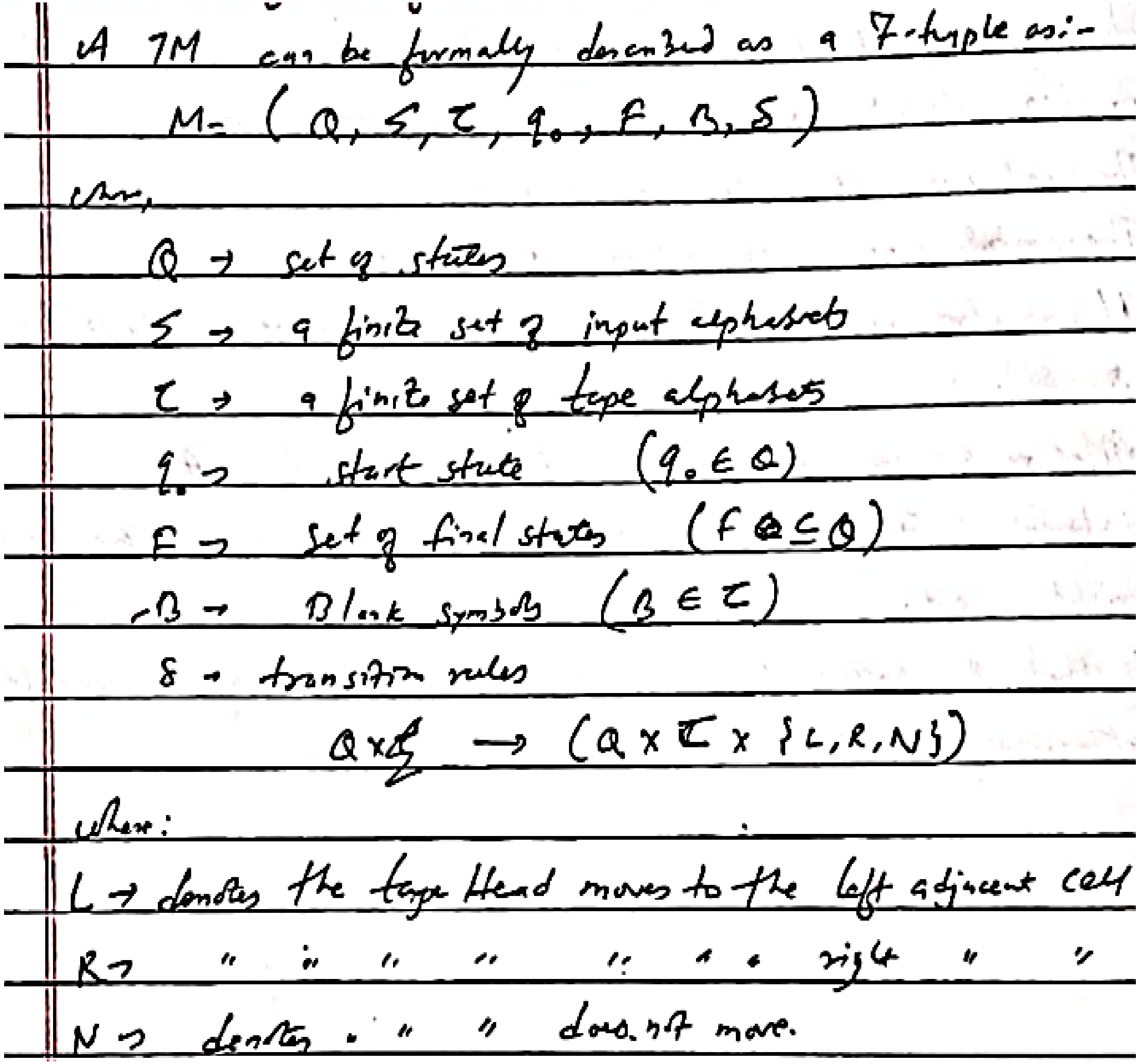


Input tape

2. The finite control and the tape head which is responsible for reading the current input symbol. The tape head can move to left to right.
3. A finite set of states through which machine has to undergo.
4. Finite set of symbols called external symbols which are used in building the logic of Turing machine.



5.3 Formal definition of TM



A TM can be formally described as a 7-tuple as:-

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

where,

Q → set of states

Σ → a finite set of input alphabets

τ → a finite set of tape alphabets

q_0 → start state $(q_0 \in Q)$

F → set of final states $(F \subseteq Q)$

B → Blank symbols $(B \in \tau)$

δ → transition rules

$$Q \times \tau \longrightarrow (Q \times \tau \times \{L, R, N\})$$

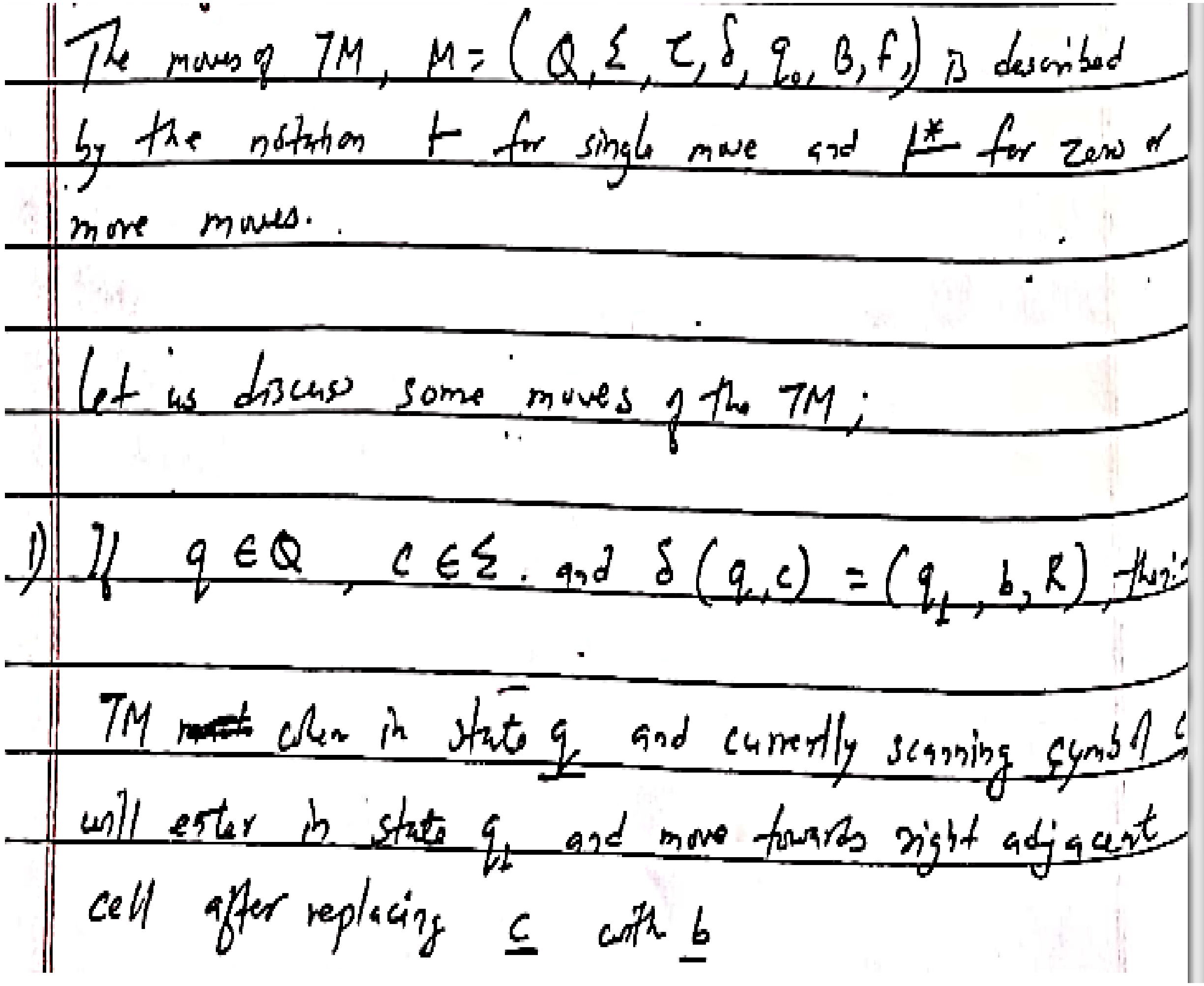
where:

L → denotes the tape Head moves to the left adjacent cell

R → " " " " " " " right " "

N → denotes " " does not move.

5.4 Moves of TM

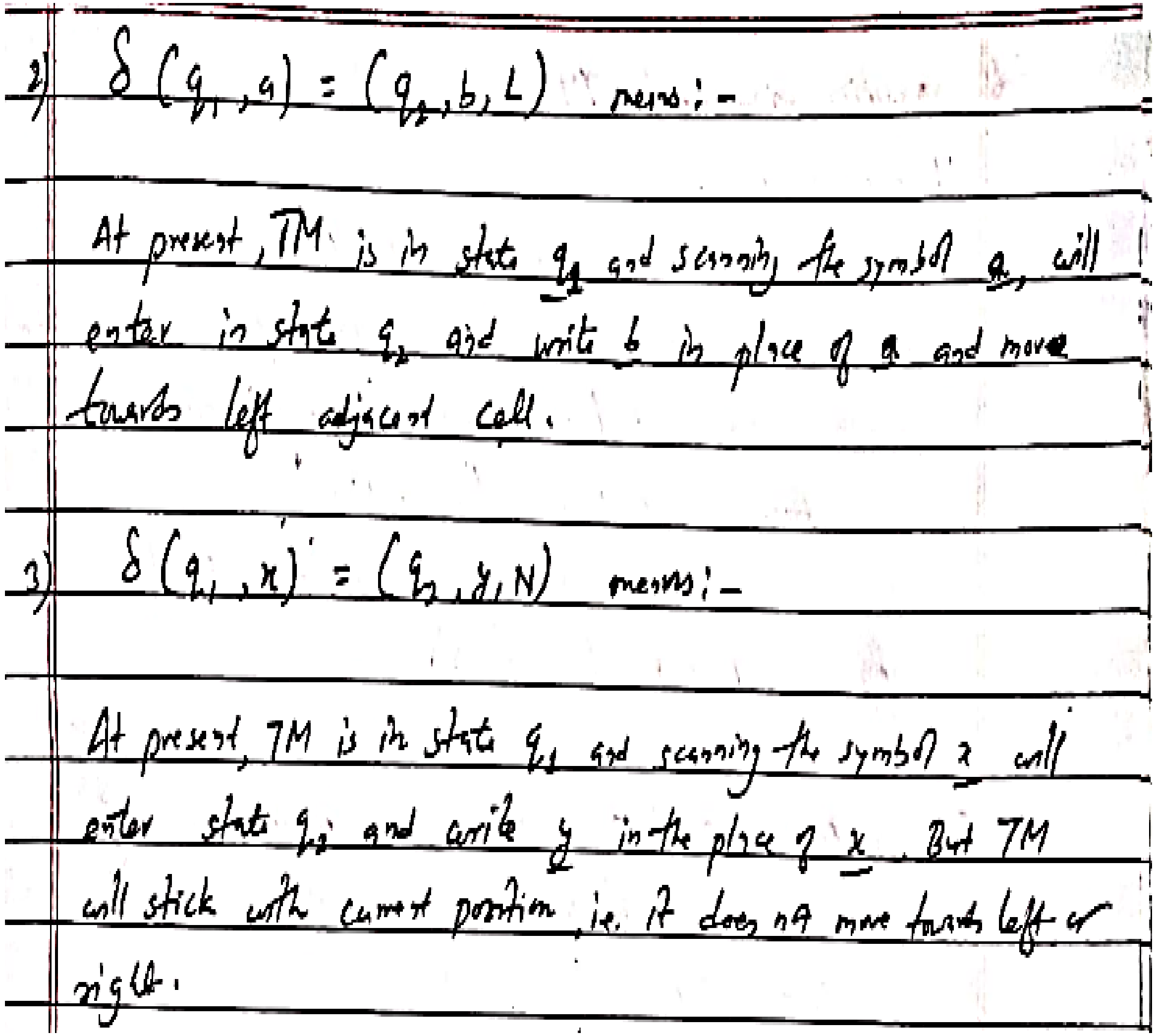


The moves of TM, $M = (Q, \Sigma, \tau, \delta, q_0, B, F)$ is described by the notation $\vdash$ for single move and $\vdash^*$ for zero or more moves.

Let us discuss some moves of the TM;

1) If $q \in Q$, $c \in \Sigma$ and $\delta(q, c) = (q_1, b, R)$, this

TM ~~must~~ when in state q and currently scanning symbol c will enter in state q_1 and move towards right adjacent cell after replacing c with b



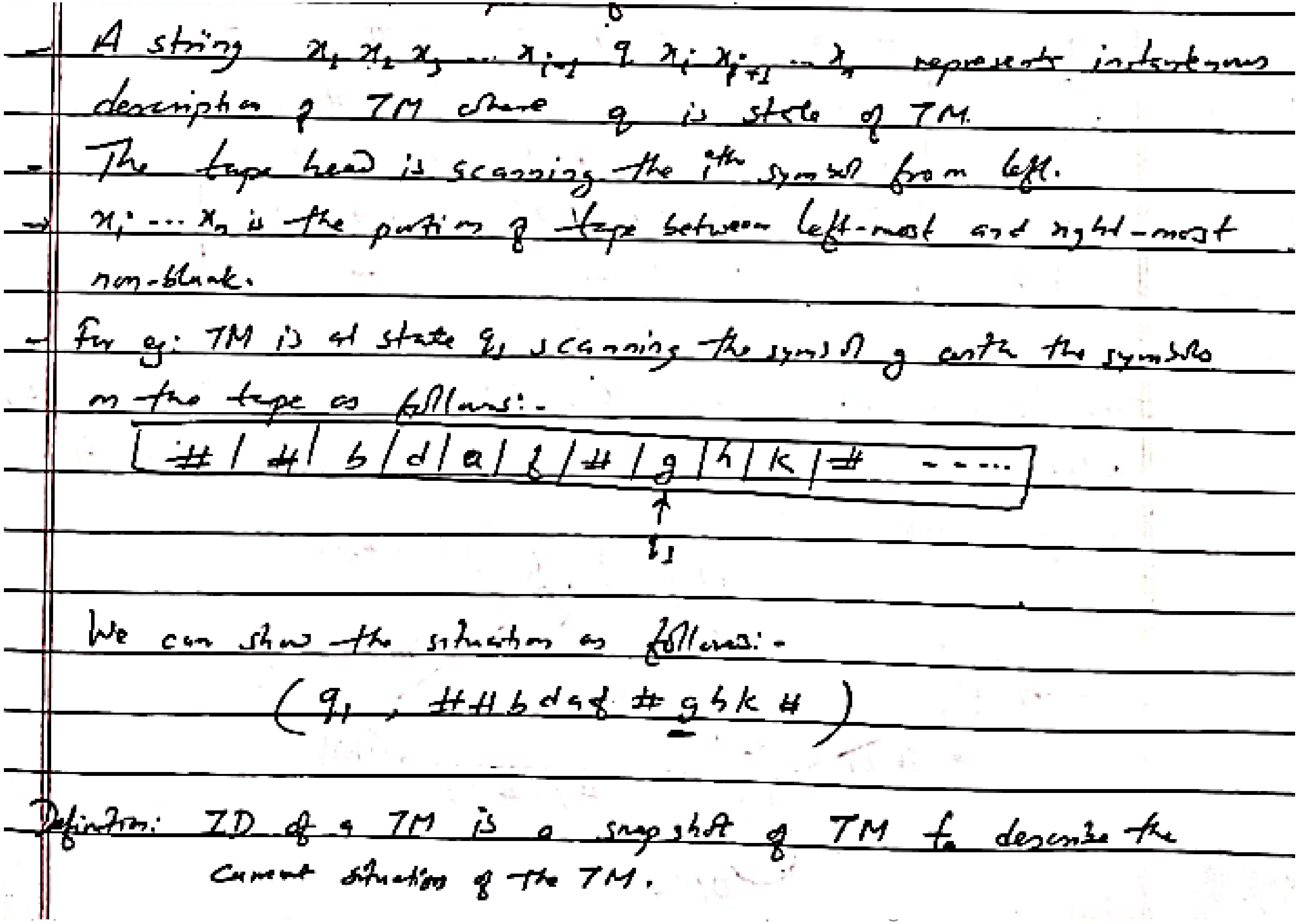
2) $\delta(q_1, a) = (q_2, b, L)$ means:-

At present, TM is in state q_1 and scanning the symbol a, will enter in state q_2 and write b in place of a and move towards left adjacent cell.

3) $\delta(q_1, x) = (q_2, y, N)$ means:-

At present, TM is in state q_1 and scanning the symbol x will enter state q_2 and write y in the place of x. But TM will stick with current position i.e. it does not move towards left or right.

5.5 Instantaneous Description for TM



- A string $x_1 x_2 x_3 \ldots x_{i-1}\, q\, x_i x_{i+1} \ldots x_n$ represents instantaneous description of TM where q is state of TM.
- The tape head is scanning the i^{th} symbol from left.
- $x_1 \ldots x_n$ is the portion of tape between left-most and right-most non-blank.
- For eg: TM is at state q_1 scanning the symbol g with the symbols on the tape as follows:-

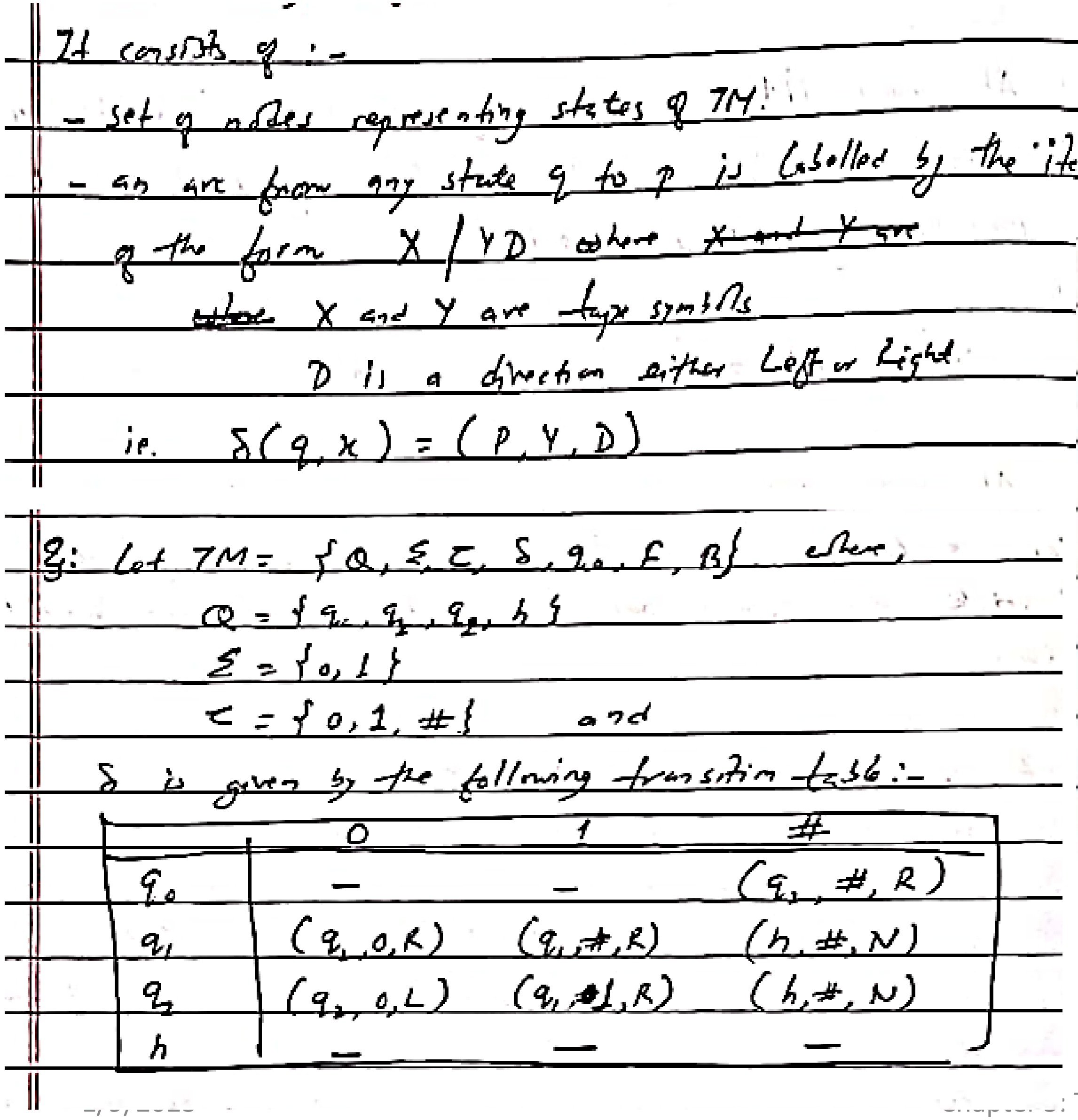
#	#	b	d	a	f	#	g	h	k	#	.....
							↑ q_1				

We can show the situation as follows:-

$$(q_1,\ \#\#bdaf\#\underline{g}hk\#)$$

Definition: ID of a TM is a snapshot of TM to describe the current situation of the TM.

5.5 Transition diagram for TM



It consists of :-

- set of nodes representing states of TM.
- an arc from any state q to p is labelled by the items of the form X / YD ~~where X and Y are~~ ~~where~~ X and Y are tape symbols D is a direction either Left or Right

i.e. $\delta(q, X) = (p, Y, D)$

Q: Let $TM = \{Q, \Sigma, \Gamma, \delta, q_0, F, B\}$ where,

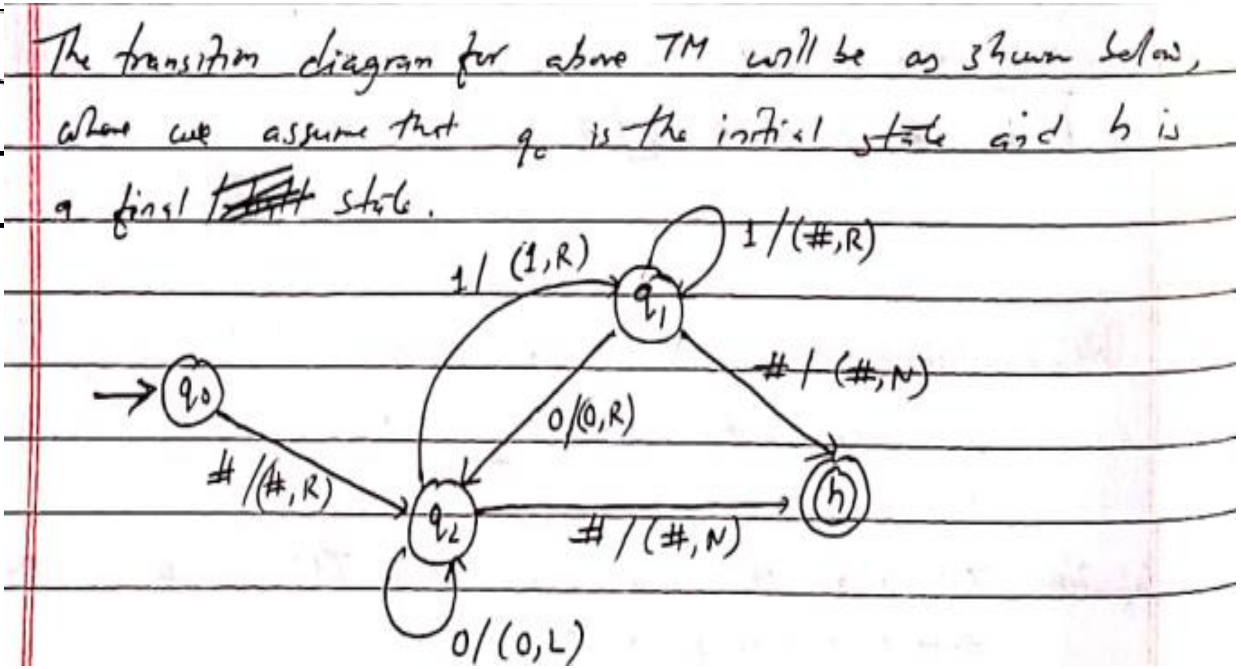
$Q = \{q_0, q_1, q_2, h\}$

$\Sigma = \{0, 1\}$

$\Gamma = \{0, 1, \#\}$ and

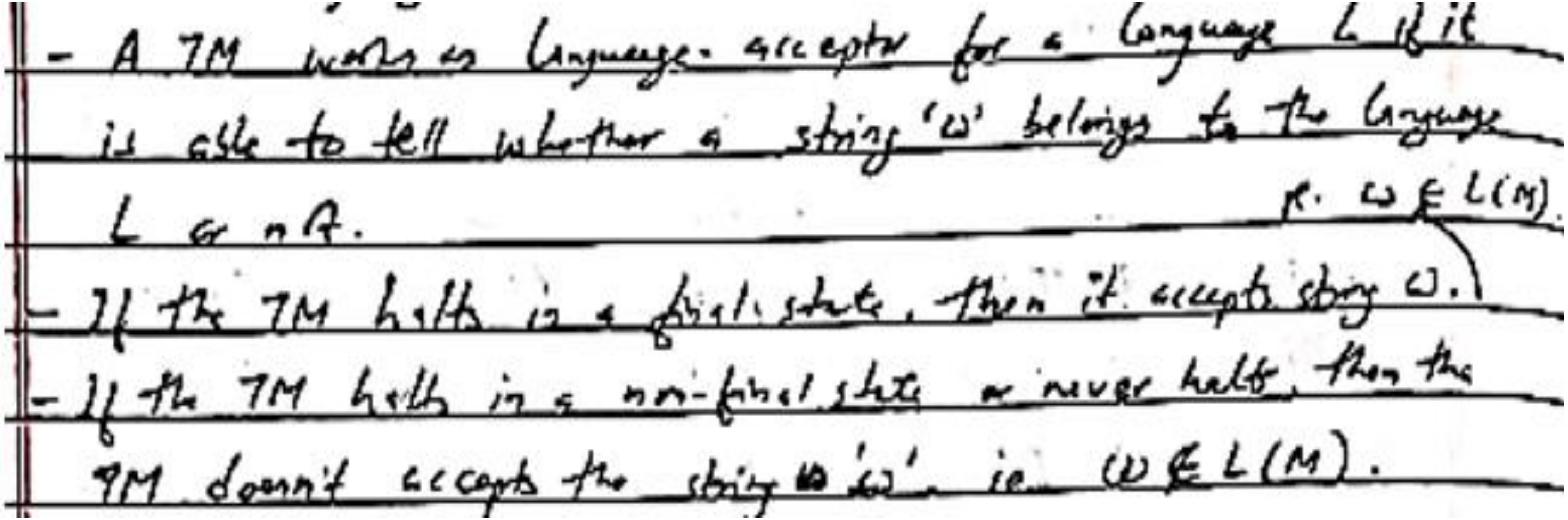
δ is given by the following transition table:-

	0	1	#
q_0	—	—	$(q_2, \#, R)$
q_1	$(q_2, 0, R)$	$(q_1, \#, R)$	$(h, \#, N)$
q_2	$(q_2, 0, L)$	$(q_1, 1, R)$	$(h, \#, N)$
h	—	—	—



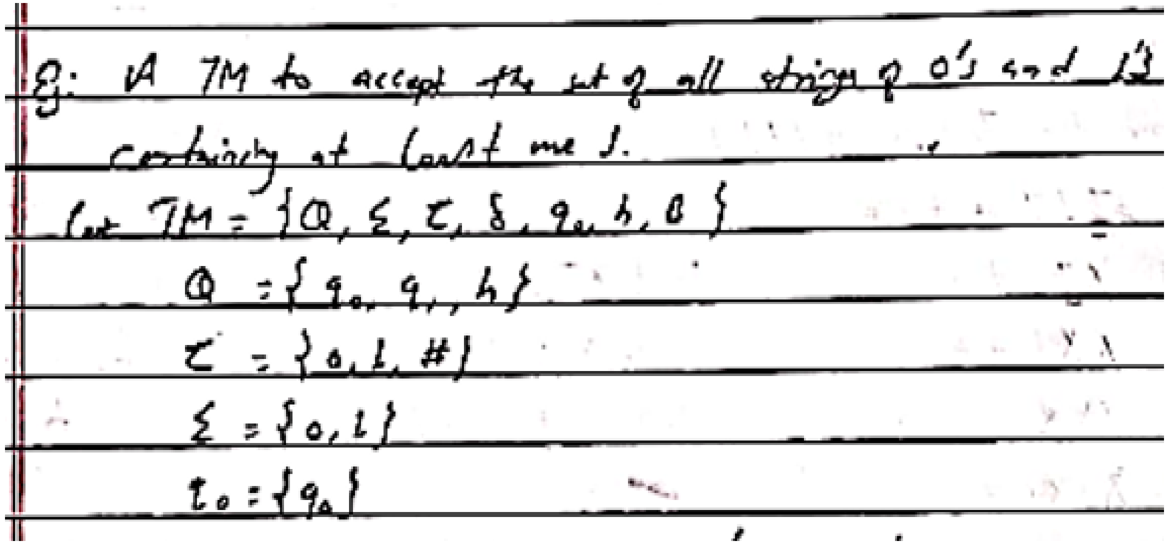
The transition diagram for above TM will be as shown below, where we assume that q_0 is the initial state and h is a final ~~state~~ state.

5.7 TM as language acceptor



- A TM works as language acceptor for a language L if it is able to tell whether a string 'ω' belongs to the language L or not. i.e. $\omega \in L(M)$.
- If the TM halts in a final state, then it accepts string ω.
- If the TM halts in a non-final state or never halts, then the TM doesn't accepts the string 'ω'. i.e. $\omega \notin L(M)$.

5.7 TM as language acceptor



Eg: A TM to accept the set of all strings of 0's and 1's containing at least one 1.

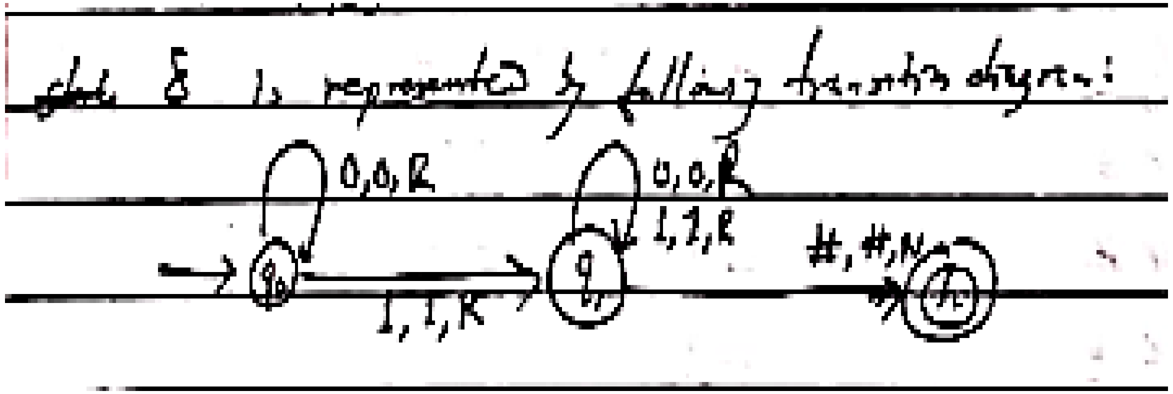
Let TM = $\{Q, \Sigma, \Gamma, \delta, q_0, h, B\}$

$Q = \{q_0, q_1, h\}$

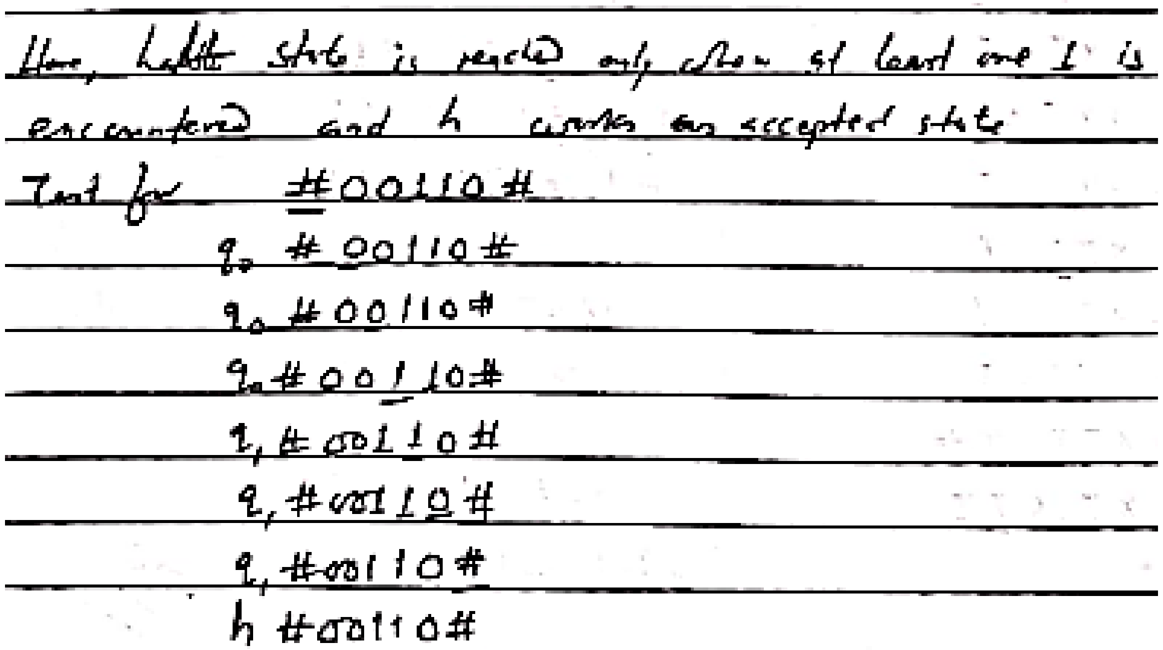
$\Gamma = \{0, 1, \#\}$

$\Sigma = \{0, 1\}$

$q_0 = \{q_0\}$



The δ is represented by following transition diagram:



Here, halt state is reached only when at least one 1 is encountered and h works as accepted state.

Test for #00110#

q_0 #00110#

q_0 #00110#

q_0 #00110#

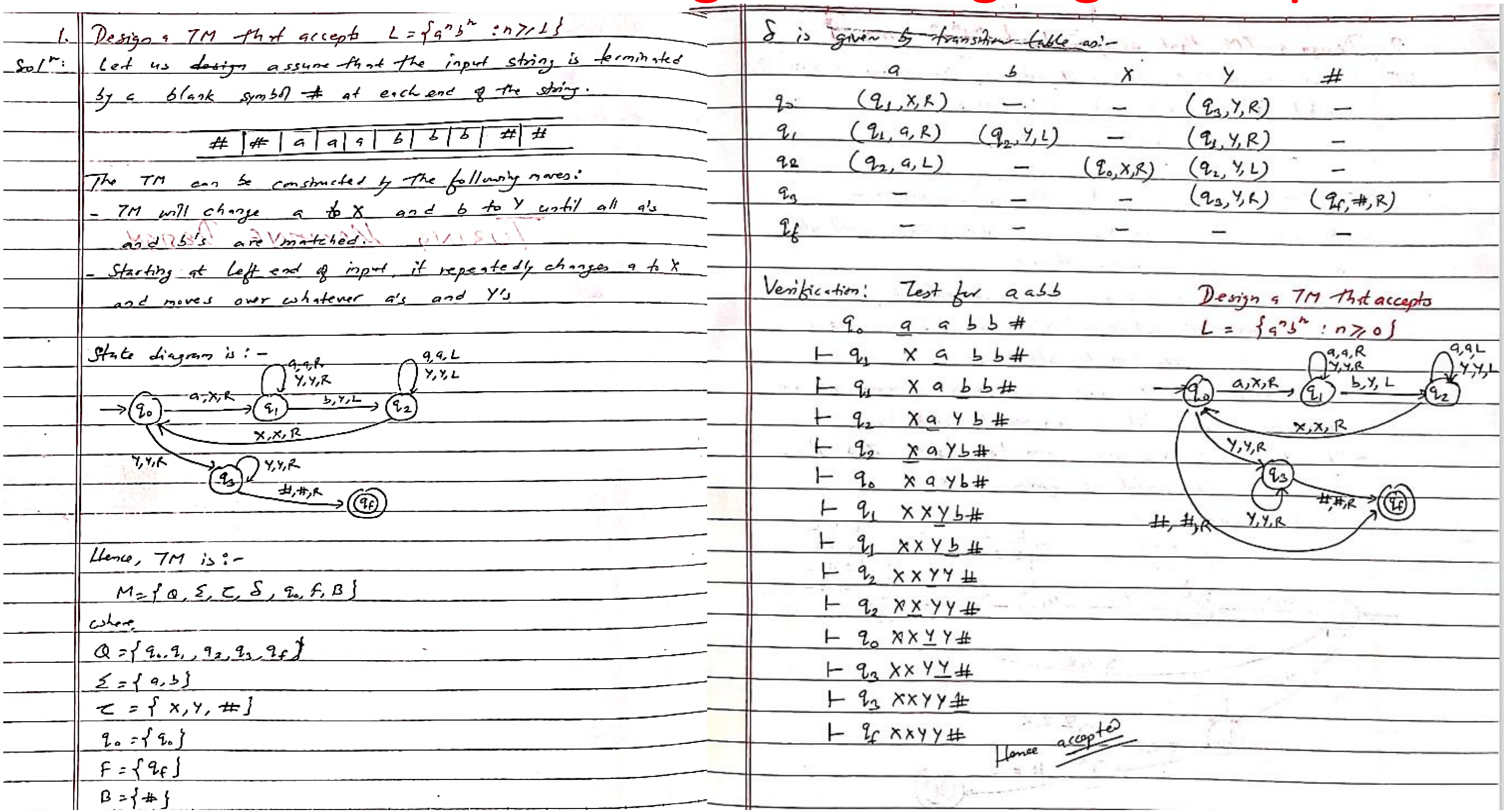
q_1 #00110#

q_1 #00110#

q_1 #00110#

h #00110#

5.8 Numerical: TM design for Language Acceptor



1. Design a TM that accept $L = \{a^n b^n : n \geq 1\}$

Sol^n: Let us ~~design~~ assume that the input string is terminated by a blank symbol # at each end of the string.

#	#	a	a	a	b	b	b	#	#

The TM can be constructed by the following moves:

- TM will change a to X and b to Y until all a's and b's are matched.
- Starting at left end of input, it repeatedly changes a to X and moves over whatever a's and Y's

State diagram is :-

a,X,R
a,a,R
Y,Y,R
b,Y,L
a,a,L
Y,Y,L
X,X,R
Y,Y,R
Y,Y,R
#,#,R
q_0
q_1
q_2
q_3
q_f

Hence, TM is :-

$M = \{Q, \Sigma, \tau, \delta, q_0, F, B\}$

where

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

$\Sigma = \{a, b\}$

$\tau = \{X, Y, \#\}$

$q_0 = \{q_0\}$

$F = \{q_f\}$

$B = \{\#\}$

δ is given by transition table as:-

	a	b	X	Y	#
q_0	(q_1, X, R)	–	–	(q_3, Y, R)	–
q_1	(q_1, a, R)	(q_2, Y, L)	–	(q_1, Y, R)	–
q_2	(q_2, a, L)	–	(q_0, X, R)	(q_2, Y, L)	–
q_3	–	–	–	(q_3, Y, R)	$(q_f, \#, R)$
q_f	–	–	–	–	–

Verification: Test for aabb

q_0 a a b b #
⊢ q_1 X a b b #
⊢ q_1 X a b b #
⊢ q_2 X a Y b #
⊢ q_2 X a Y b #
⊢ q_0 X a Y b #
⊢ q_1 X X Y b #
⊢ q_1 X X Y b #
⊢ q_2 X X Y Y #
⊢ q_2 X X Y Y #
⊢ q_0 X X Y Y #
⊢ q_3 X X Y Y #
⊢ q_3 X X Y Y #
⊢ q_f X X Y Y #

Hence accepted

Design a TM that accepts $L = \{a^n b^n : n \geq 0\}$

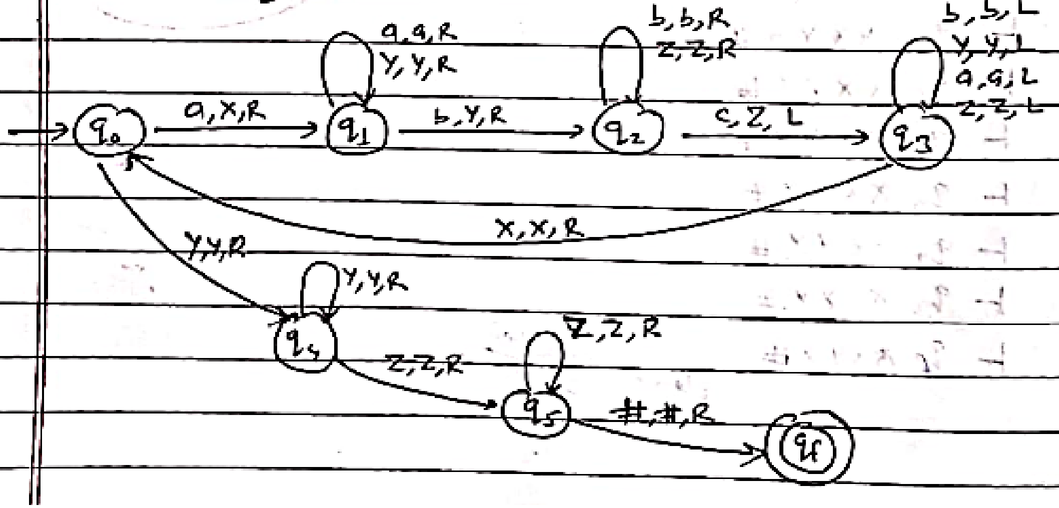
2. Design a TM that accepts $L = \{a^n b^n c^n : n \geq 1\}$

Sol^n: Let us assume that the input string is terminated by a blank symbol #, at each end of the string.

| a | a | b | b | c | c |

The TM can be constructed by the following moves:-

- TM will change a to X, b to Y and c to Z until all a's, b's and c's are matched.
- Starting at left end of input, it repeatedly changes a to X and moves over whatever a's and Y's ~~[illegible]~~ it sees until it reaches to state q_1.
- It changes b to Y and moves over whatever b's and Z's it sees until it reaches to state q_2.
- It changes c to Z and moves left over Z's, Y's, b's and a's until it finds an X.
- State diagram is :-



Hence, TM is: $M = \{Q, \Sigma, \tau, \delta, q_0, F, B\}$

where, $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_f\}$ $\quad q_0 = \{q_0\}$

$\Sigma = \{a, b, c\}$ $\quad F = \{q_f\}$

$\tau = \{X, Y, Z, \#\}$ $\quad B = \{\#\}$

δ is given by:-

	a	b	c	X	Y	Z	#
q_0	(q_1, X, R)	–	–	–	(q_4, Y, R)	–	–
q_1	(q_1, a, R)	(q_2, Y, R)	–	–	(q_1, Y, R)	–	–
q_2	–	(q_2, b, R)	(q_3, Z, L)	–	–	(q_2, Z, R)	–
q_3	(q_3, a, L)	(q_3, b, L)	–	(q_0, X, R)	(q_3, Y, L)	(q_3, Z, L)	–
q_4	–	–	–	–	(q_4, Y, R)	(q_5, Z, R)	–
q_5	–	–	–	–	–	(q_5, Z, R)	$(q_f, \#, R)$
q_f	–	–	–	–	–	–	–

Verification: for aabbcc

$q_0\ \underline{a}abbcc\#$

$\vdash q_1\ X\underline{a}bbcc\#$

$\vdash q_1\ Xa\underline{b}bcc\#$

$\vdash q_2\ XaY\underline{b}cc\#$

$\vdash q_2\ XaYb\underline{c}c\#$

$\vdash q_3\ XaY\underline{b}Zc\#$

$\vdash q_3\ Xa\underline{Y}bZc\#$

$\vdash q_3\ X\underline{a}YbZc\#$

$\vdash q_3\ \underline{X}aYbZc\#$

$\vdash q_0\ X\underline{a}YbZc\#$

$\vdash q_1\ XX\underline{Y}bZc\#$

$\vdash q_1\ XXY\underline{b}Zc\#$

$\vdash q_2\ XXYY\underline{Z}c\#$

$\vdash q_2\ XXYYZ\underline{c}\#$

$\vdash q_3\ XXYY\underline{Z}Z\#$

$\vdash q_3\ XXY\underline{Y}ZZ\#$

$\vdash q_3\ XX\underline{Y}YZZ\#$

$\vdash q_3\ X\underline{X}YYZZ\#$

$\vdash q_0\ XX\underline{Y}YZZ\#$

$\vdash q_4\ XXY\underline{Y}ZZ\#$

$\vdash q_4\ XXYY\underline{Z}Z\#$

$\vdash q_5\ XXYYZ\underline{Z}\#$

$\vdash q_5\ XXYYZZ\underline{\#}$

$\vdash q_f\ XXYYZZ\#$ acapted

3. Design a TM that accept $L = \omega \in \{a,b\}^*$ and no of a's and no of b's are equal.

Sol^n: Clearly the language accepts strings like ab, ba, aabb, abab, abba, aabbba, abbbaa, ----

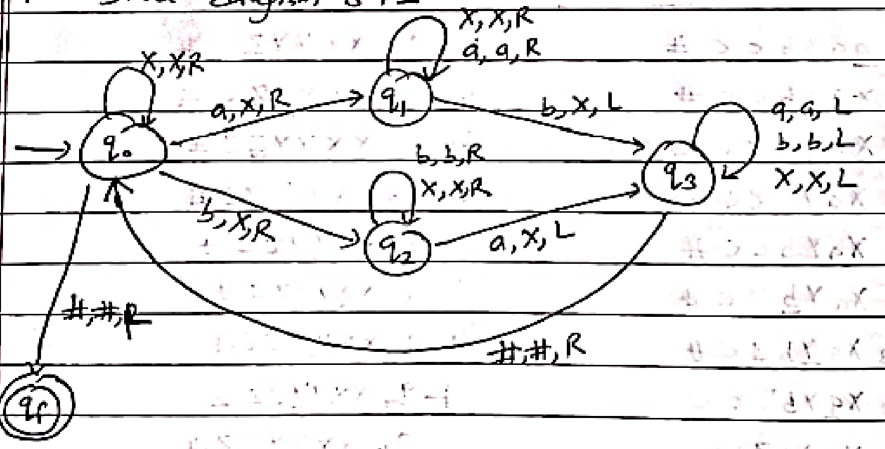
Let us assume that the input string is terminated by a blank symbol #, at each end of the string.

#	a	a	b	b	b	a	#

TM is constructed by the following moves:-

- TM will change a to X, and finds b and changes it to X.
- Repeatedly move left and right, each pair of a and b are replaced with X.

The state diagram is:-



Hence, TM is: $M = \{Q, \Sigma, \tau, \delta, q_0, F, B\}$

where $Q = \{q_0, q_1, q_2, q_3, q_f\}$ $\quad q_0 = \{q_0\}$

$\Sigma = \{a, b\}$ $\quad F = \{q_f\}$

$\tau = \{X, \#\}$ $\quad B = \{\#\}$

δ is given by:-

	a	b	X	#
q_0	(q_1, X, R)	(q_2, X, R)	(q_0, X, R)	$(q_f, \#, L)$
q_1	(q_1, a, R)	(q_3, X, L)	(q_1, X, R)	—
q_2	(q_3, X, L)	(q_2, b, R)	(q_2, X, R)	—
q_3	(q_3, a, L)	(q_3, b, L)	(q_3, X, L)	$(q_0, \#, R)$
q_f	—	—	—	—

Verification: For aabbba:

$q_0 \# \underline{a}\, a\, b\, b\, b\, a \#$

$\vdash q_1 \# X\, a\, b\, b\, b\, a\, \#$

$\vdash q_1 \# X\, a\, \underline{b}\, b\, b\, a\, \#$

$\vdash q_3 \# X\, a\, X\, b\, b\, a\, \#$

$\vdash q_3 \# \underline{X}\, a\, X\, b\, b\, a\, \#$

$\vdash q_3\, \underline{\#}\, X\, a\, X\, b\, b\, a \#$

$\vdash q_0\, \#\, \underline{X}\, a\, X\, b\, b\, a \#$

$\vdash q_0\, \#\, X\, \underline{a}\, X\, b\, b\, a\, \#$

$\vdash q_1\, \#\, X\, X\, \underline{X}\, b\, b\, a\, \#$

$\vdash q_1\, \#\, X\, X\, X\, \underline{b}\, b\, a\, \#$

$\vdash q_3\, \#\, X\, X\, \underline{X}\, X\, \underline{b}\, a\, \#$

$\vdash q_3\, \#\, X\, X\, X\, X\, b\, a\, \#$

$\vdash q_3\, \#\, \underline{X}\, X\, X\, X\, b\, a\, \#$

$\vdash q_3\, \underline{\#}\, X\, X\, X\, X\, b\, a\, \#$

$\vdash q_0\, \#\, \underline{X}\, X\, X\, X\, b\, a\, \#$

$\vdash q_0\, \#\, X\, \underline{X}\, X\, X\, b\, a\, \#$

$\vdash q_0\, \#\, X\, X\, \underline{X}\, X\, b\, a\, \#$

$\vdash q_0\, \#\, X\, X\, X\, \underline{X}\, b\, a\, \#$

$\vdash q_0\, \#\, X\, X\, X\, X\, \underline{b}\, a\, \#$

$\vdash q_2\, \#\, X\, X\, X\, X\, X\, \underline{a}\, \#$

$\vdash q_3\, \#\, X\, X\, X\, X\, \underline{X}\, X\, \#$

$\vdash^* q_3\, \underline{\#}\, X\, X\, X\, X\, X\, X\, \#$

$\vdash q_0\, \#\, \underline{X}\, X\, X\, X\, X\, X\, \#$

$\vdash^* q_0\, \#\, X\, X\, X\, X\, X\, \underline{X}\, \#$

$\vdash q_0\, \#\, X\, X\, X\, X\, X\, X\, \underline{\#}$

$\vdash q_f\, \#\, X\, X\, X\, X\, X\, X\, \#$

Hence accepted

4. Design a TM for $L = \{ wcw \mid w \in \{a,b\}^* \}$

Soln: Clearly the string generated by the language L is like:

abcab , abbcabb , abaabcabaab, ---

Let us assume that the input string has blank symbol # at both the extreme ends.

#	a	b	a	a	b	c	a	b	a	a	b	#

We design the TM as in below fashion:-

- TM scans a in the first half and changes it into X, and finds the corresponding a in the second half and changes a to X.
- TM scans b in first half and changes to Y, and correspondingly changes b in second half to Y.
- Thus all a are changed into X and b into Y.

The ~~state diagram~~ TM is:

$M = \{ Q, \Sigma, \Gamma, \delta, q_0, F, B \}$

where, $Q = \{ q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_f \}$

$\Sigma = \{a, b\}$

$\Gamma = \{ X, \# \}$

$q_0 = \{q_0\}$

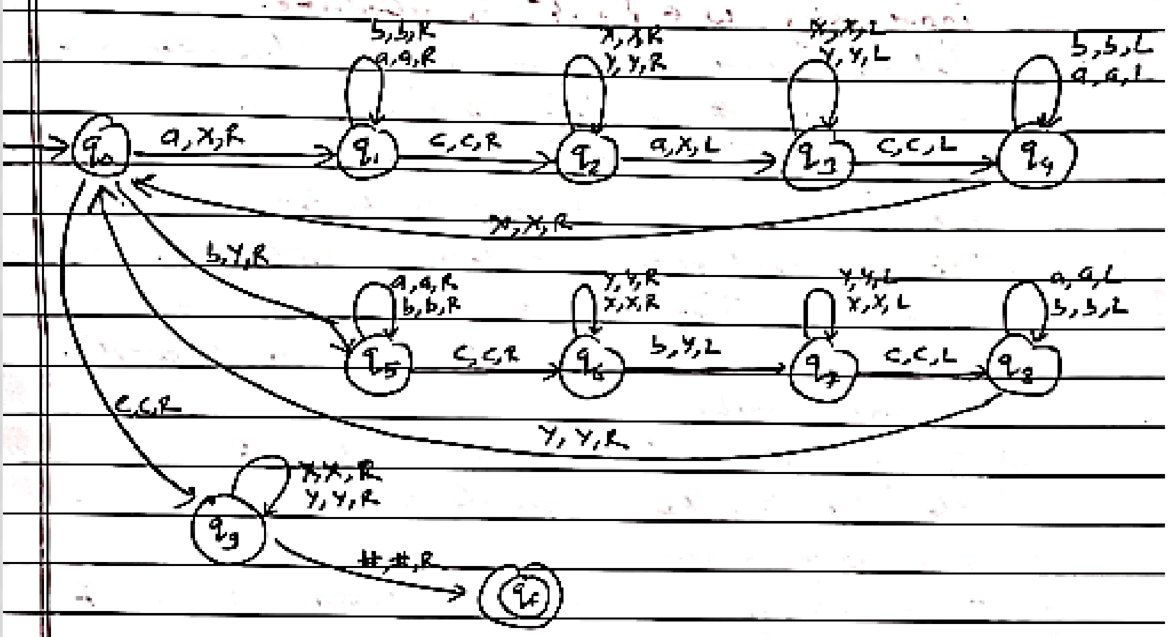
$F = \{q_f\}$

$B = \{\#\}$

δ is given by:

	a	b	c	#
q_0	--	--	--	--
q_1	--	--	--	--
..	--	--	--	--
..	--	--	--	--
q_f				

The state diagram is:-



Verification: for string abbcabb

q_0 #abbcabb#

$\vdash q_1$ # X b b c a b b #

$\vdash q_1$ # X b b c a b b #

$\vdash q_1$ # X b b c a b b #

$\vdash q_2$ # X b b c a b b #

$\vdash q_3$ # X b b c X b b #

$\vdash^* q_f$ # X Y Y c X Y Y #. (Hence accepted)

5. Design a TM to ~~decide~~ decide whether or not any input string $\omega \in \{a,b\}^*$ is palindrome.

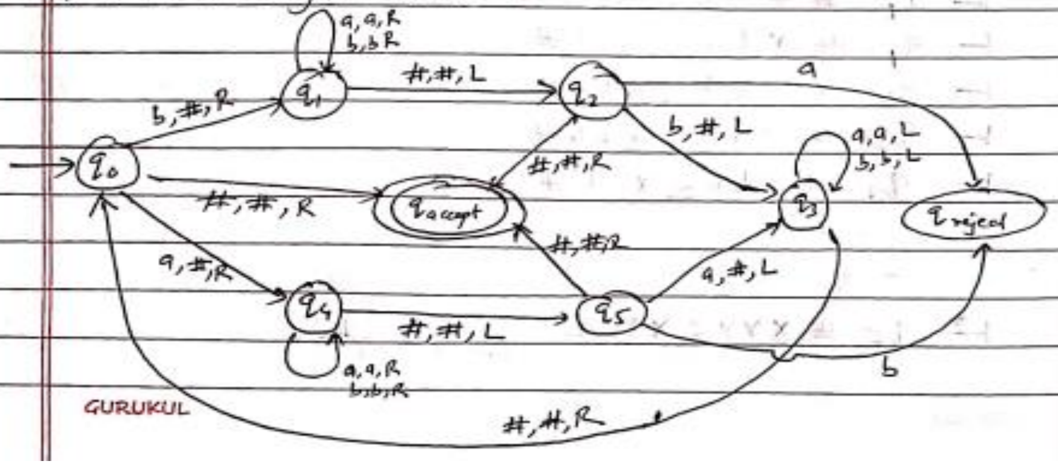
Sol^n: Let us assume that input string is terminated by a blank symbol #, at right ~~[illegible]~~ end of the string.

| a | b | b | a | # |

The TM can be constructed by the following moves:-

- The tape head ~~move~~ reads leftmost symbol of ω, replaces it by #. Then the tape head moves to the right most symbol and test whether it is equal to the leftmost symbol.
- If they are equal, then Right most symbol is deleted, then the tape head moves to the new left-most symbol and above process is repeated.
- If corresponding left most and right-most cells are not equal, TM enters the reject state and stops.
- TM enters accept state as soon as string currently stored in tape is empty.

The state diagram is:-



Here TM is :-

$M = \{Q, \Sigma, \tau, \delta, q_0, F, B\}$

Where,

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_{accept}, q_{reject}\}$

$\Sigma = \{a, b\}$

$\tau = \{a, b, \#\}$

$q_0 = \{q_0\}$

$F = \{q_{accept}\}$

$B = \{\#\}$

δ is given by:

	a	b	#
q_0	$(q_4, \#, R)$	$(q_1, \#, R)$	q_{accept}
q_1	(q_1, a, R)	(q_1, b, R)	$(q_2, \#, L)$
q_2	–	$(q_3, \#, L)$	q_{accept}
q_3	$(q_3\ a, L)$	(q_3, b, L)	$(q_0, \#, R)$
q_4	(q_4, a, R)	(q_4, b, R)	$(q_5, \#, L)$
q_5	$(q_3, \#, L)$		$q_{accept})$
q_{accept}	–	–	–

Verification : test for abba

$q_0 : \underline{a}bba\#$

$\vdash q_4\ \#\underline{b}ba\#$

$\vdash q_4\ \#b\underline{b}a\#$

$\vdash q_4\ \#bb\underline{a}\#$

$\vdash q_4\ \#bba\underline{\#}$

$\vdash q_5\ \#bb\underline{a}\#$

$\vdash q_3\ \#b\underline{b}\#\#$

$\vdash q_3\ \#\underline{b}b\#\#$

$\vdash q_3\ \underline{\#}bb\#\#$

$\vdash q_0\ \#\underline{b}b\#\#$

$\vdash q_1\ \#\#\underline{b}\#\#$

$\vdash q_1\ \#\#b\underline{\#}\#$

$\vdash q_2\ \#\#\underline{b}\#\#$

$\vdash q_3\ \#\underline{\#}\#\#\#$

$\vdash q_0\ \#\#\underline{\#}\#\#$

$\vdash q_{accept}\ \#\#\#\underline{\#}\#$

Since, the TM replaces all symbol by # and enters q_{accept} state. The input string is palindrome.

Also check for ab#

ab#

$q_0\ \underline{a}b\#$

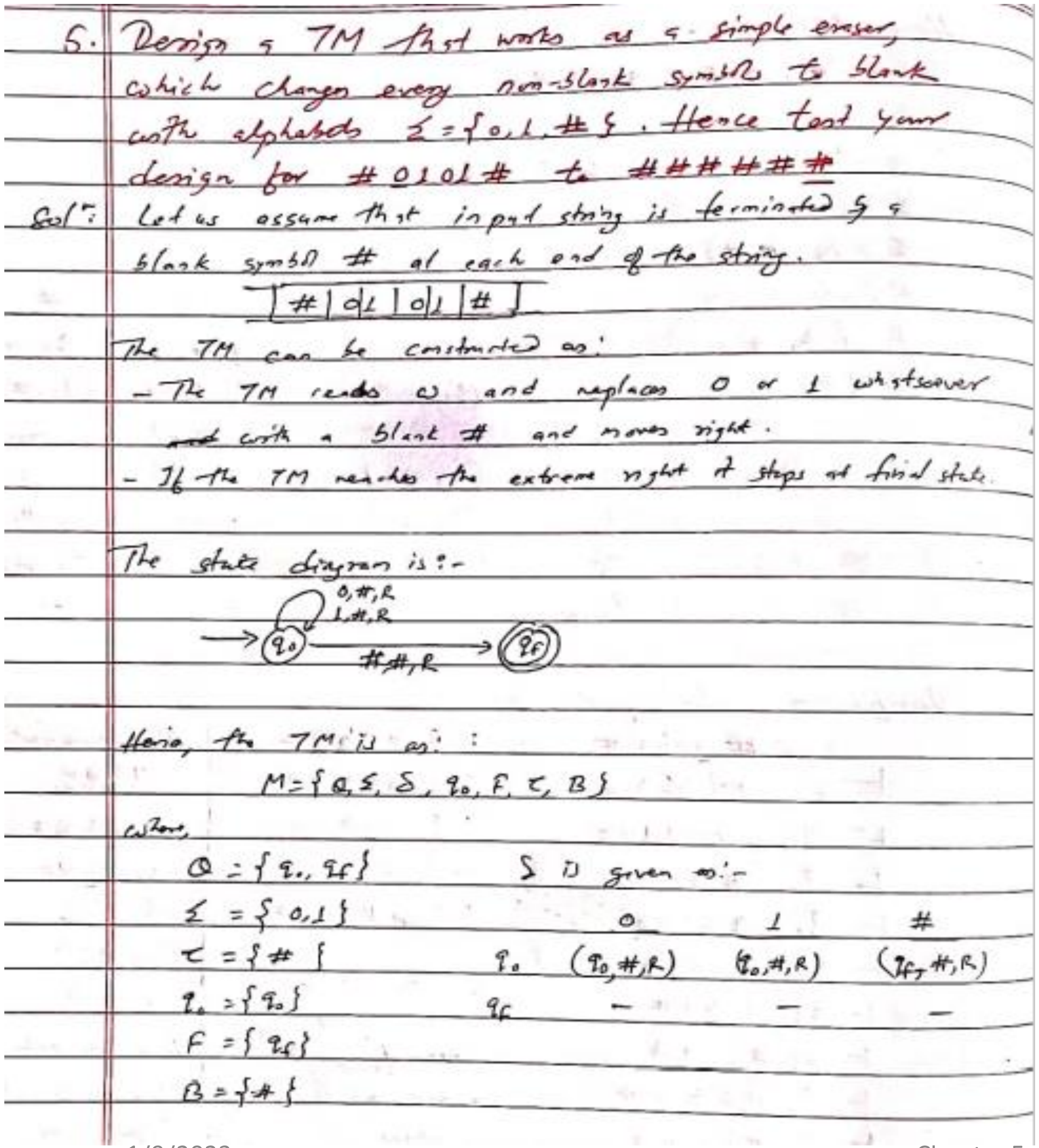
$q_4\ \#\underline{b}\#$

$q_4\ \#b\underline{\#}$

$q_5\ \#\underline{b}\#$

$q_{rej}\ \#b\#$

Since TM enters q_{reject} state, the input string is not palindrome.



5. Design a TM that works as a simple eraser, which changes every non-blank symbol to blank with alphabets $\Sigma = \{0, 1, \#\}$. Hence test your design for #0̲101# to ######

Sol^n: Let us assume that input string is terminated by a blank symbol # at each end of the string.

| # | 0/1 | 0/1 | # |

The TM can be constructed as:

- The TM reads w and replaces 0 or 1 whatsoever ~~and~~ with a blank # and moves right.
- If the TM reaches the extreme right it stops at final state.

The state diagram is:-

Hence, the TM is as: :

$M = \{Q, \Sigma, \delta, q_0, F, \tau, B\}$

where,

$Q = \{q_0, q_f\}$

$\Sigma = \{0, 1\}$

$\tau = \{\#\}$

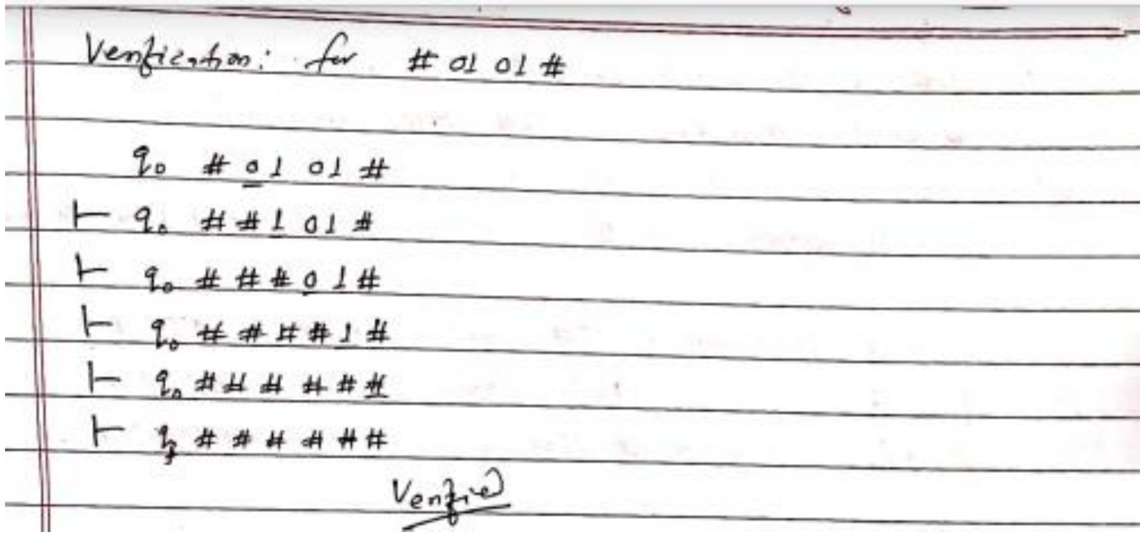
$q_0 = \{q_0\}$

$F = \{q_f\}$

$B = \{\#\}$

δ is given as:-

	0	1	#
q_0	$(q_0, \#, R)$	$(q_0, \#, R)$	$(q_f, \#, R)$
q_f	–	–	–



Verification: for #0101#

$q_0\ \#\underline{0}101\#$

$\vdash q_0\ \#\#\underline{1}01\#$

$\vdash q_0\ \#\#\#\underline{0}1\#$

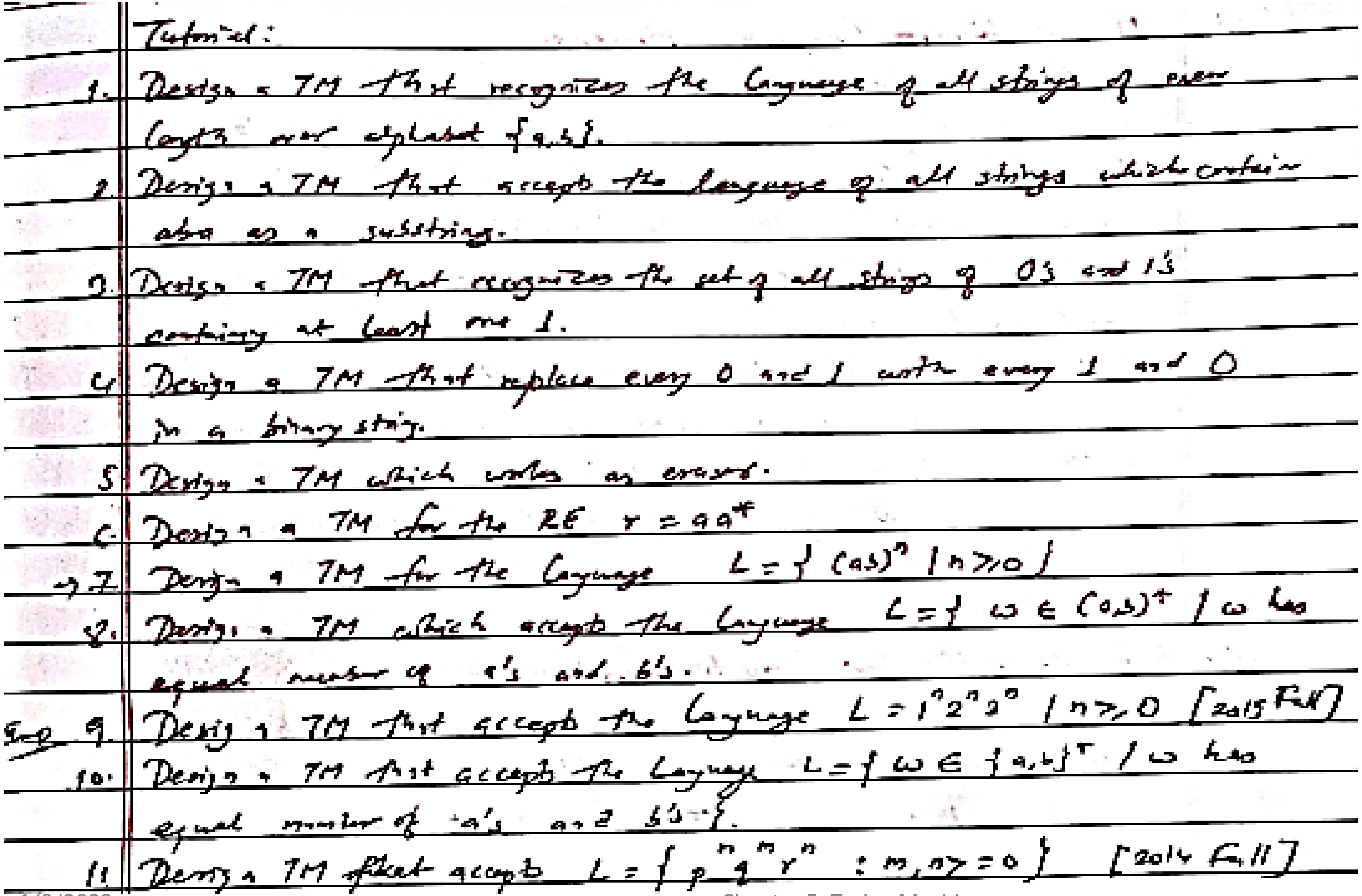
$\vdash q_0\ \#\#\#\#1\#$

$\vdash q_0\ \#\#\#\#\#\#$

$\vdash q_f\ \#\#\#\#\#\#$

Verified

Assignment:



Tutorial:

1. Design a TM that recognizes the language of all strings of even length over alphabet {a,b}.
2. Design a TM that accepts the language of all strings which contain aba as a substring.
3. Design a TM that recognizes the set of all strings of 0's and 1's containing at least one 1.
4. Design a TM that replace every 0 and 1 with every 1 and 0 in a binary string.
5. Design a TM which works as erasor.
6. Design a TM for the RE $r = aa^*$

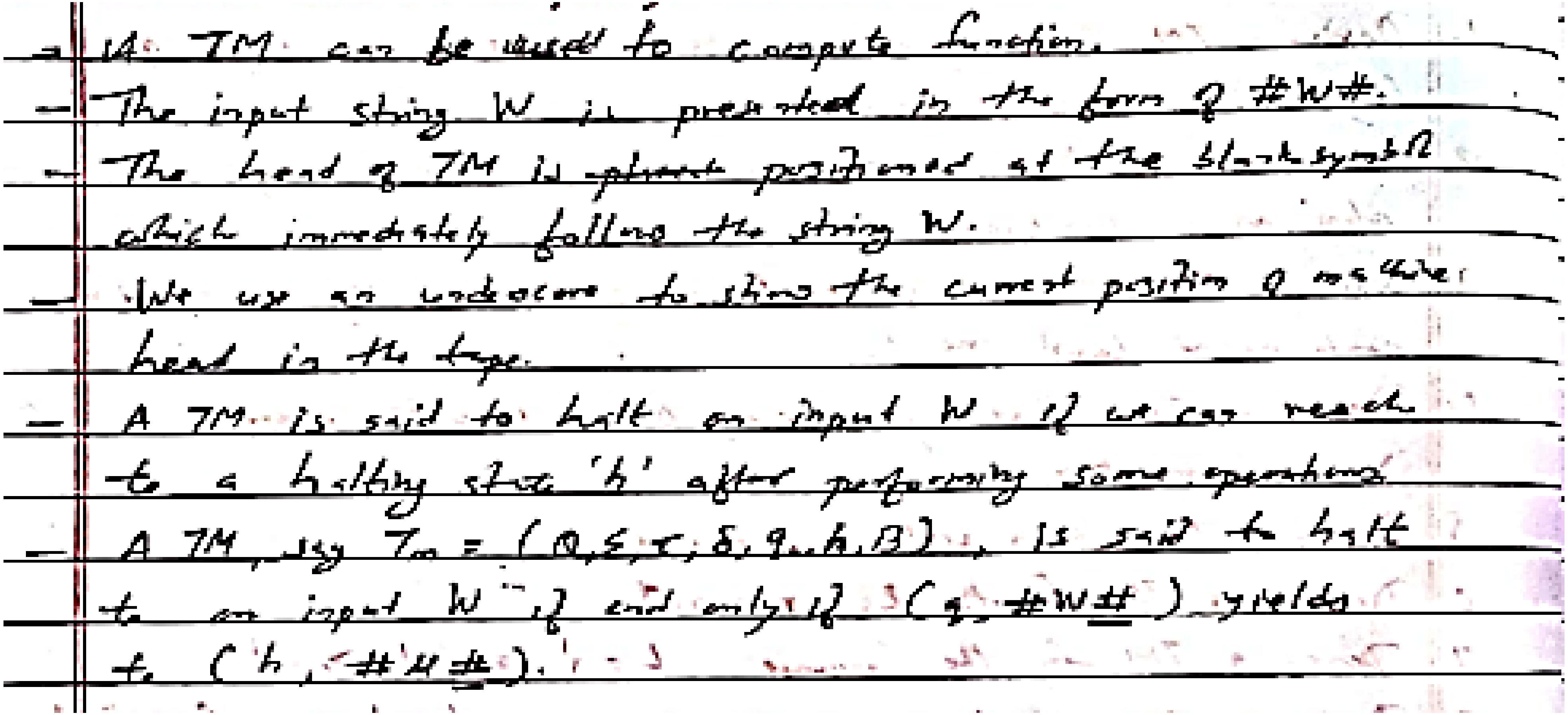
→ 7. Design a TM for the language $L = \{ (ab)^n \mid n \geq 0 \}$

8. Design a TM which accepts the language $L = \{ \omega \in (a,b)^* \mid \omega$ has equal number of a's and b's.

Imp 9. Design a TM that accepts the language $L = 1^n 2^n 3^n \mid n \geq 0$ [2015 Fall]

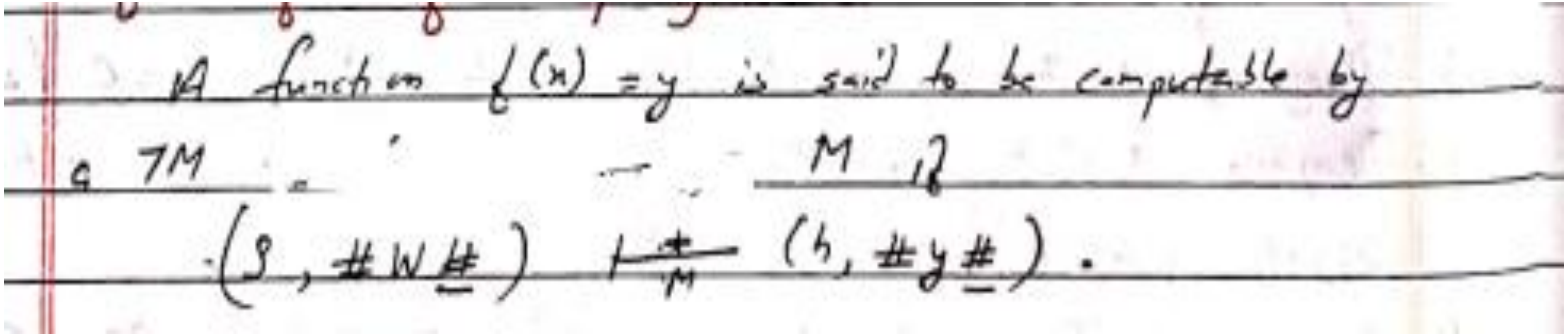
10. Design a TM that accepts the language $L = \{ \omega \in \{a,b\}^* \mid \omega$ has equal number of a's and b's $\}$.
11. Design a TM that accepts $L = \{ p^n q^m r^n : m, n >= 0 \}$ [2014 Fall]

5.9 TM for computing functions



- A TM can be used to compute function.
- The input string W is presented in the form of #W#.
- The head of TM is ~~placed~~ positioned at the blank symbol which immediately follows the string W.
- We use an underscore to show the current position of machine head in the tape.
- A TM is said to halt on input W if we can reach to a halting state 'h' after performing some operations.
- A TM, say $T_m = (Q, \Sigma, \Gamma, \delta, q_0, h, B)$, is said to halt to an input W if and only if $(q_0, \#W\underline{\#})$ yields to $(h, \#W\underline{\#})$.

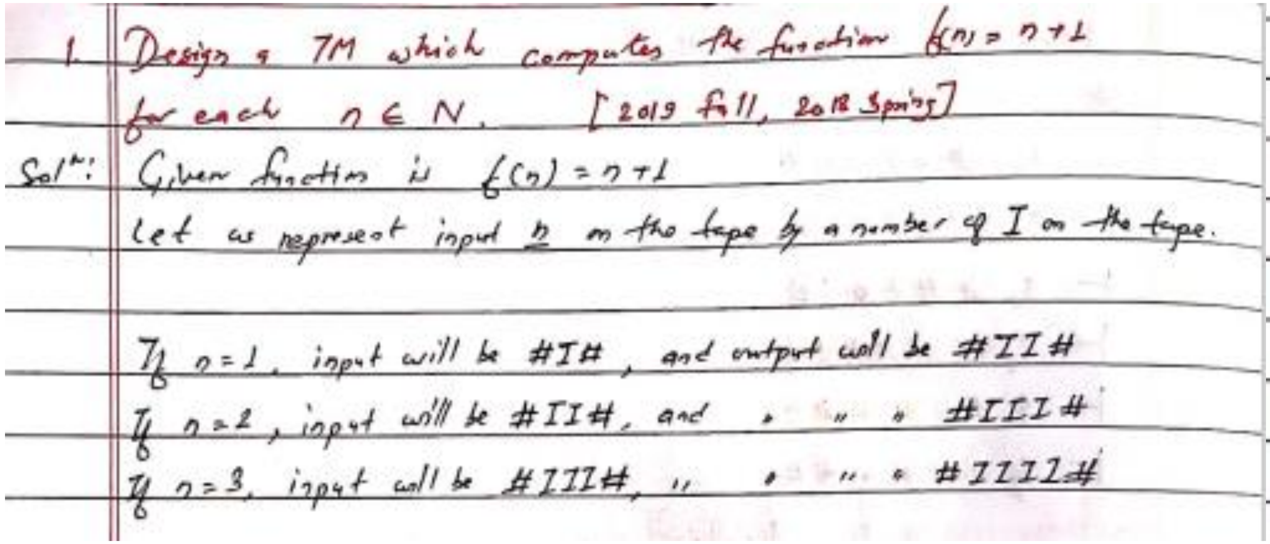
5.10 Definition of TM for Computing function



A function $f(x) = y$ is said to be computable by a TM M if

$$(s, \#w\underline{\#}) \vdash^{*}_{M} (h, \#y\underline{\#}).$$

5.11 Numerical: TM for Computing Function



1. Design a TM which computes the function $f(n) = n+1$ for each $n \in N$. [2019 fall, 2018 Spring]

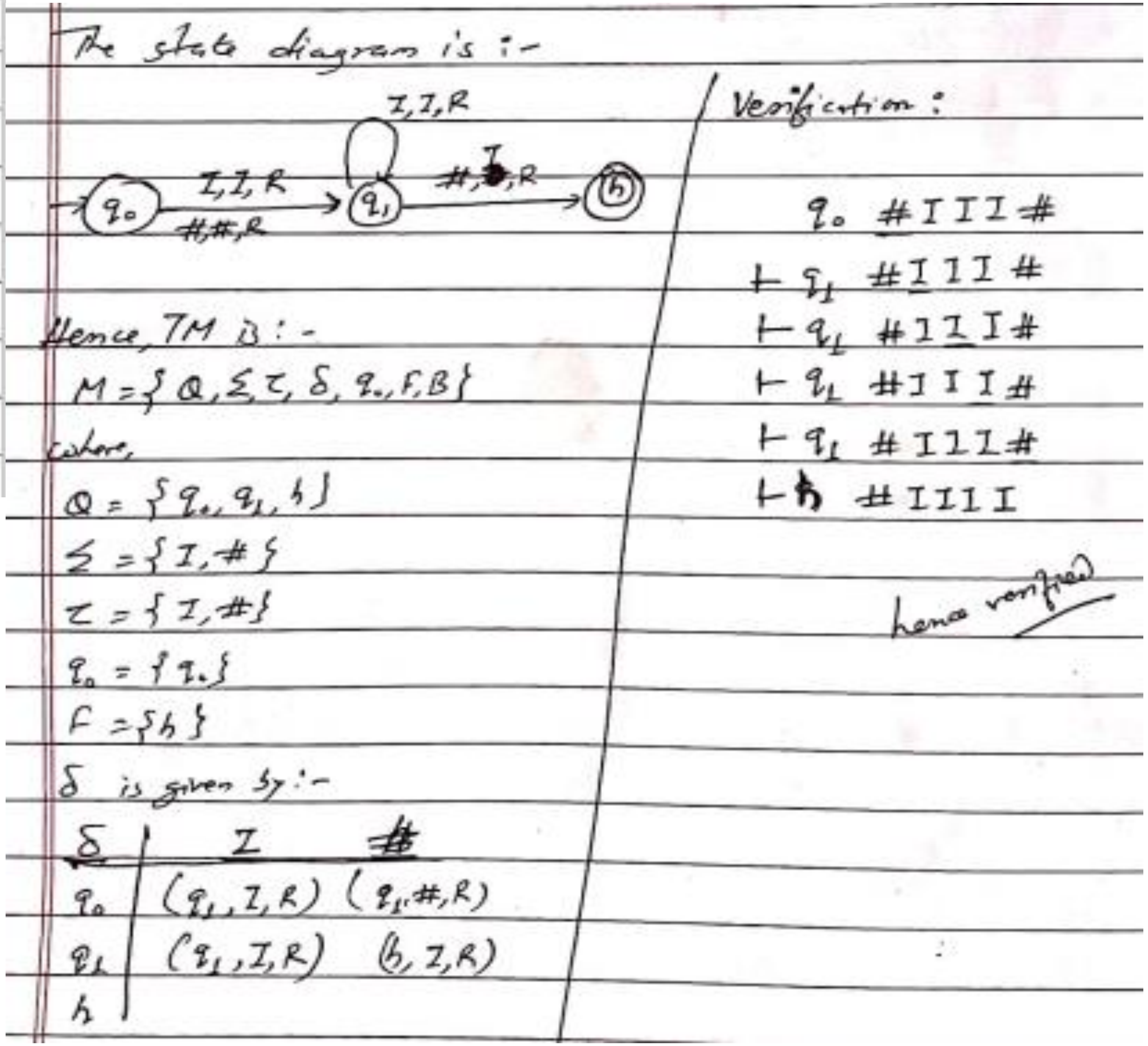
Solⁿ: Given function is $f(n) = n+1$

Let us represent input n on the tape by a number of I on the tape.

If $n=1$, input will be #I#, and output will be #II#

If $n=2$, input will be #II#, and " " " #III#

If $n=3$, input will be #III#, " " " " #IIII#



The state diagram is :-

q₀
q₁
h
I, I, R
#, #, R
I, I, R
#, I, R

Hence, TM is :-

$M = \{Q, \Sigma, \tau, \delta, q_0, F, B\}$

where,

$Q = \{q_0, q_1, h\}$

$\Sigma = \{I, \#\}$

$\tau = \{I, \#\}$

$q_0 = \{q_0\}$

$F = \{h\}$

δ is given by :-

δ	I	#
q_0	(q_1, I, R)	$(q_1, \#, R)$
q_1	(q_1, I, R)	(h, I, R)
h		

Verification :

$q_0 \; \underline{\#}III\#$

$\vdash q_1 \; \#\underline{I}II\#$

$\vdash q_1 \; \#I\underline{I}I\#$

$\vdash q_1 \; \#II\underline{I}\#$

$\vdash q_1 \; \#III\underline{\#}$

$\vdash h \; \#IIII$

hence verified

3. Design a TM for computing a function $f(w) = W\#W$

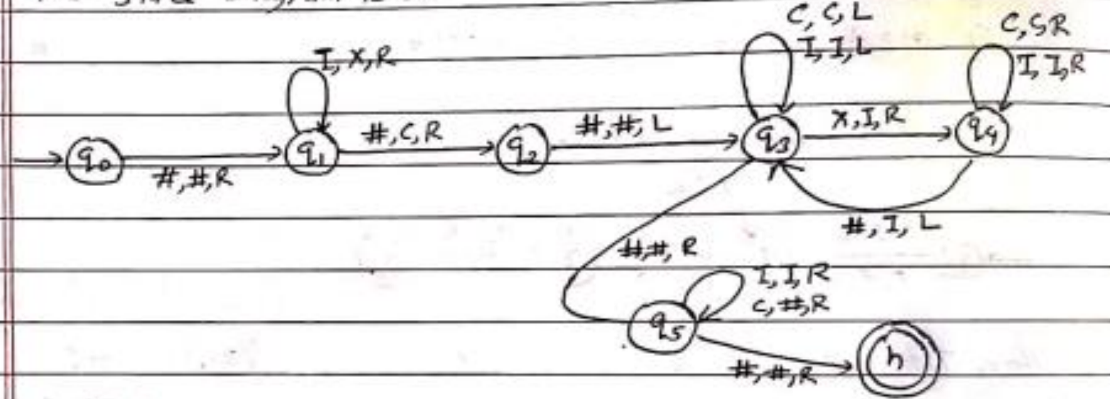
Sol^n: Here,

We have to design a TM that takes W and gives output W#W.

Let us assume, W as #III#,

then, the output should be #III#III#

The state diagram is:-



Hence, the TM is:-

$M = \{Q, \Sigma, \Gamma, \delta, q_0, F, B\}$

where,

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, h\}$

$\Sigma = \{I, \#\}$

$\Gamma = \{X, I, C, \#\}$

$B = \{\#\}$

$q_0 = \{q_0\}$

$F = \{h\}$

δ is given by:-

	I	#	X	C
q_0	–	$(q_1, \#, R)$	–	–
q_1	(q_1, X, R)	(q_2, C, R)	–	–
q_2	–	$(q_3, \#, L)$	–	–
q_3	(q_3, I, L)	$(q_5, \#, R)$	(q_4, I, R)	(q_3, C, L)
q_4	(q_4, I, R)	(q_3, I, L)	–	(q_4, C, R)
q_5	(q_5, I, R)	$(h, \#, R)$	–	$(q_5, \#, R)$
h	–	–	–	–

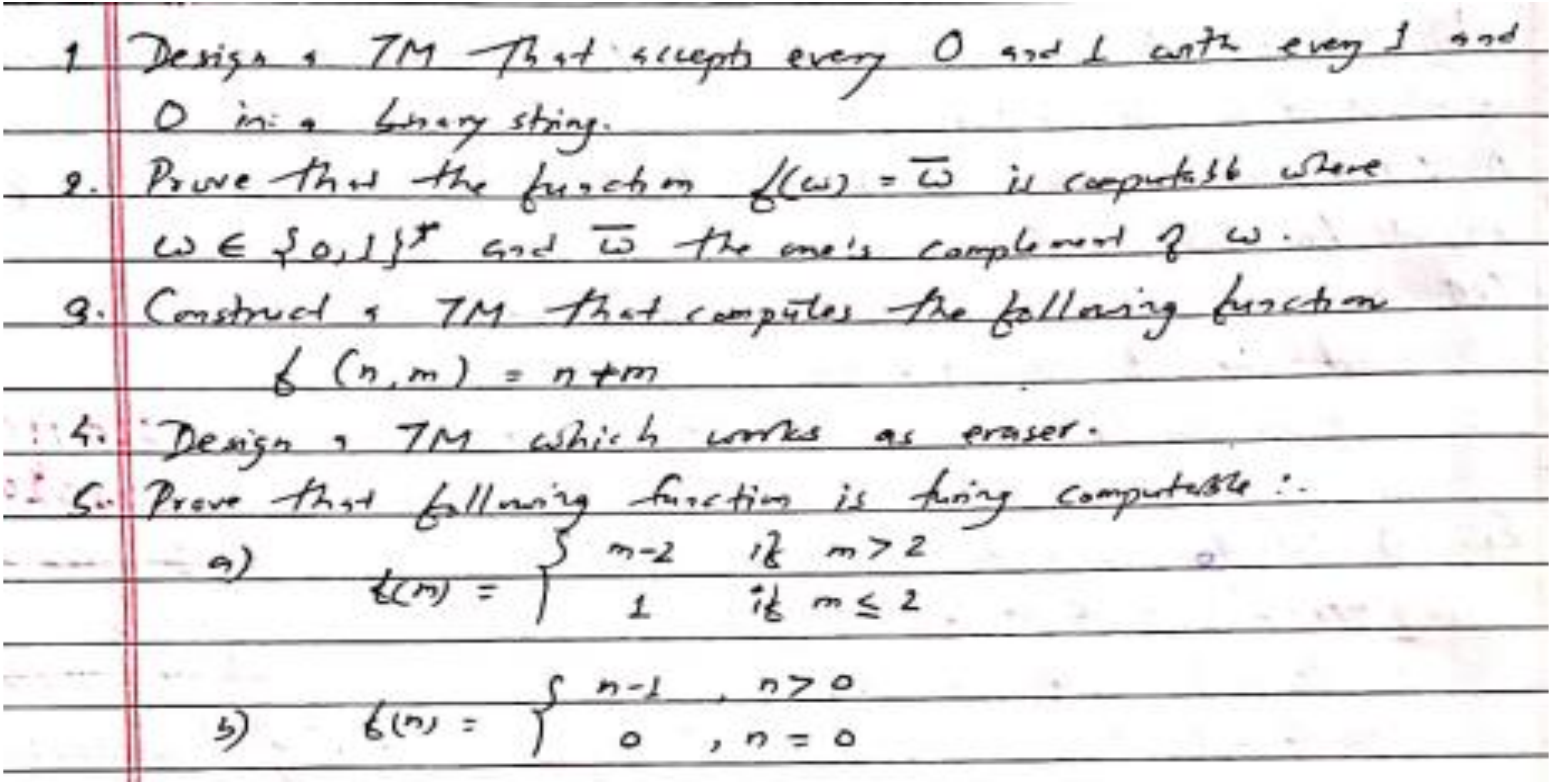
Verification: for #II#

1 $\vdash q_0\ \underline{\#}II\#\#\#\#$
2 $\vdash q_1\ \#\underline{I}I\#\#\#\#$
3 $\vdash q_1\ \#X\underline{I}\#\#\#\#$
4 $\vdash q_1\ \#XX\underline{\#}\#\#\#$
5 $\vdash q_3\ \#X\underline{X}C\#\#\#$
6 $\vdash q_4\ \#XI\underline{C}\#\#\#$
7 $\vdash q_4\ \#XIC\underline{\#}\#\#$
8 $\vdash q_3\ \#XI\underline{C}I\#\#$
9 $\vdash q_3\ \#X\underline{I}CI\#\#$
10 $\vdash q_3\ \#\underline{X}ICI\#\#$
11 $\vdash q_4\ \#I\underline{I}CI\#\#$
12 $\vdash q_4\ \#II\underline{C}I\#\#$
13 $\vdash q_4\ \#IIC\underline{I}\#\#$
14 $\vdash q_4\ \#IICI\underline{\#}\#$
15 $\vdash q_3\ \#IIC\underline{I}I\#$
16 $\vdash q_3\ \#II\underline{C}II\#$
17 $\vdash q_3\ \#I\underline{I}CII\#$
18 $\vdash q_3\ \#\underline{I}ICII\#$
19 $\vdash q_3\ \underline{\#}IICII\#$
20 $\vdash q_5\ \#\underline{I}ICII\#$
21 $\vdash q_5\ \#I\underline{I}CII\#$
22 $\vdash q_5\ \#II\underline{C}II\#$
23 $\vdash q_5\ \#II\#\underline{I}I\#$
24 $\vdash q_5\ \#II\#I\underline{I}\#$
25 $\vdash q_5\ \#II\#II\underline{\#}$
26 $\vdash h\ \#II\#II\#$

hence accepted

Note: Take #I# for verification to minimize time consumption

Assignment:



1 Design a TM that accepts every 0 and 1 with every 1 and 0 in a binary string.

2. Prove that the function $f(\omega) = \overline{\omega}$ is computable where $\omega \in \{0,1\}^*$ and $\overline{\omega}$ the one's complement of ω.

3. Construct a TM that computes the following function

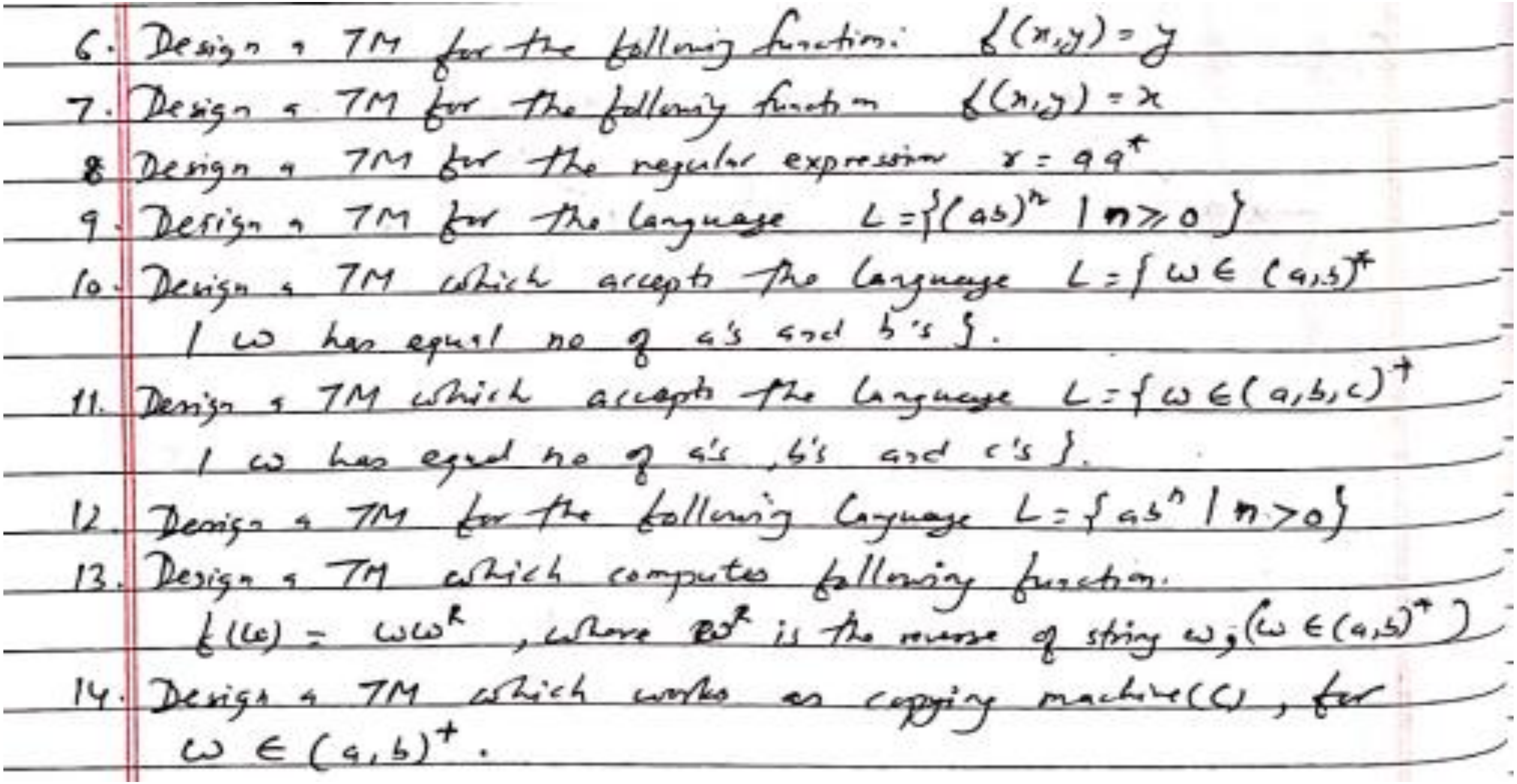
$$f(n, m) = n + m$$

4. Design a TM which works as eraser.

5. Prove that following function is turing computable :-

a) $$f(m) = \begin{cases} m-2 & \text{if } m > 2 \\ 1 & \text{if } m \leq 2 \end{cases}$$

b) $$f(n) = \begin{cases} n-1 , & n > 0 \\ 0 , & n = 0 \end{cases}$$



6. Design a TM for the following function: $f(x,y) = y$

7. Design a TM for the following function $f(x,y) = x$

8 Design a TM for the regular expression $r = aa^*$

9. Design a TM for the language $L = \{(ab)^n \mid n \geq 0\}$

10. Design a TM which accepts the language $L = \{ \omega \in (a,b)^+$
$\mid \omega$ has equal no of a's and b's $\}$.

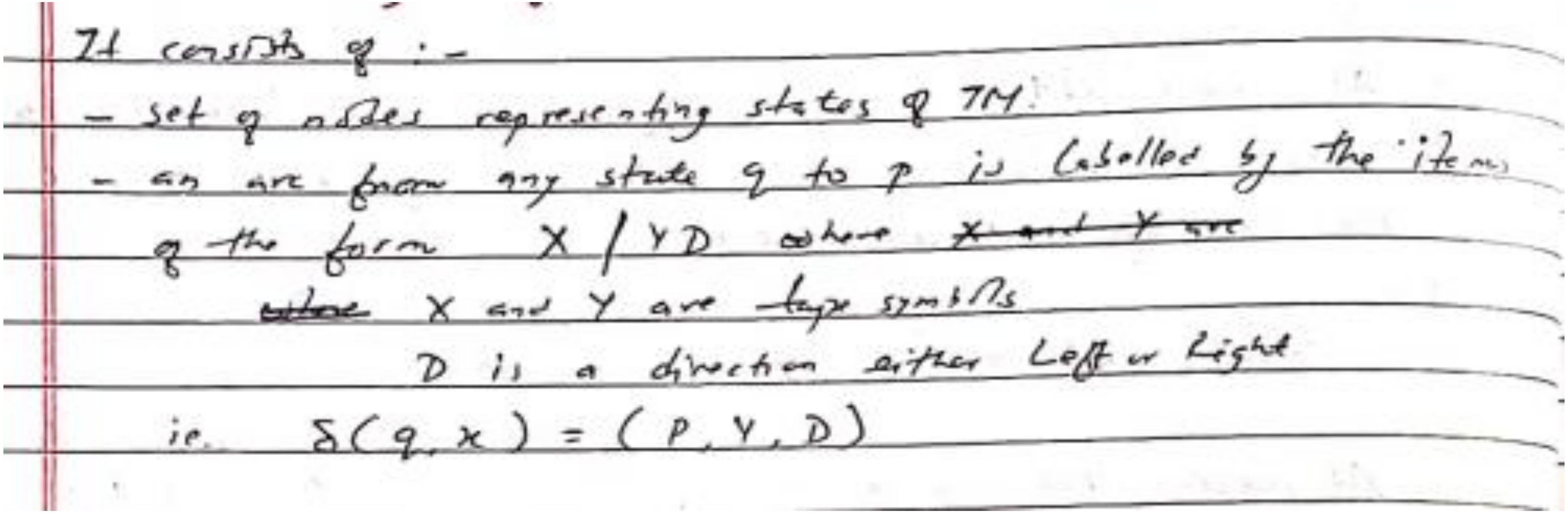
11. Design a TM which accepts the language $L = \{ \omega \in (a,b,c)^+$
$\mid \omega$ has equal no of a's, b's and c's $\}$.

12. Design a TM for the following language $L = \{ab^n \mid n > 0\}$

13. Design a TM which computes following function.
$f(\omega) = \omega\omega^R$, where ω^R is the reverse of string ω, $(\omega \in (a,b)^+)$

14. Design a TM which works as copying machine(C), for
$\omega \in (a,b)^+$.

5.12 TM as Transducer



It consists of :-

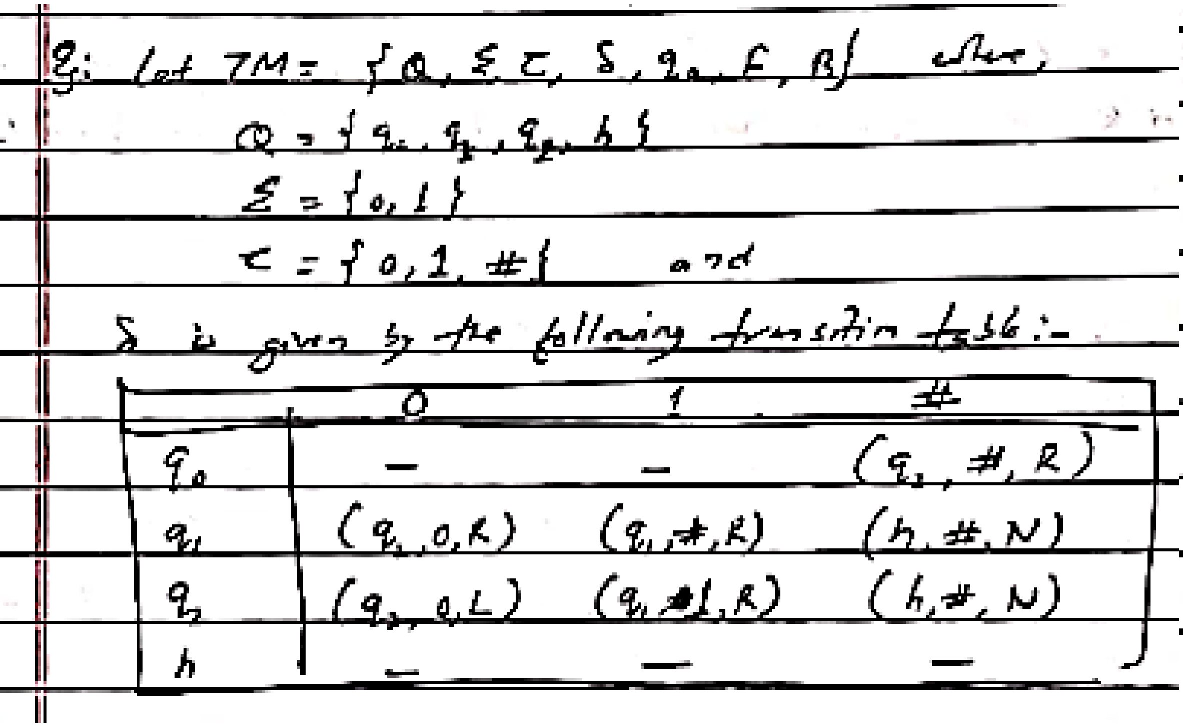
- set of nodes representing states of TM.
- an arc from any state q to p is labelled by the items of the form X / Y D where ~~X and Y are~~

~~where~~ X and Y are tape symbols

D is a direction either Left or Right

ie. $\delta(q, x) = (p, Y, D)$

5.12 TM as Transducer



Q: Let TM = {Q, Σ, Γ, δ, q_0, F, B} where,

Q = {q_0, q_1, q_2, h}

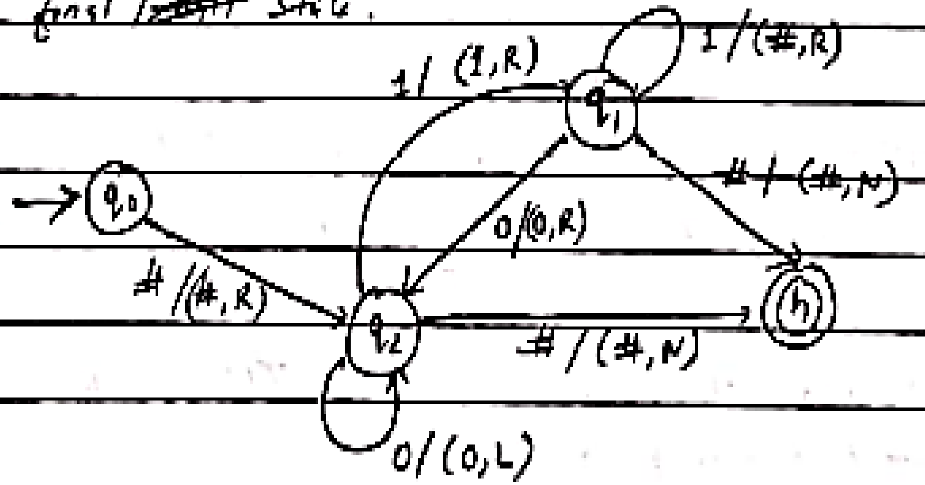
Σ = {0, 1}

Γ = {0, 1, #} and

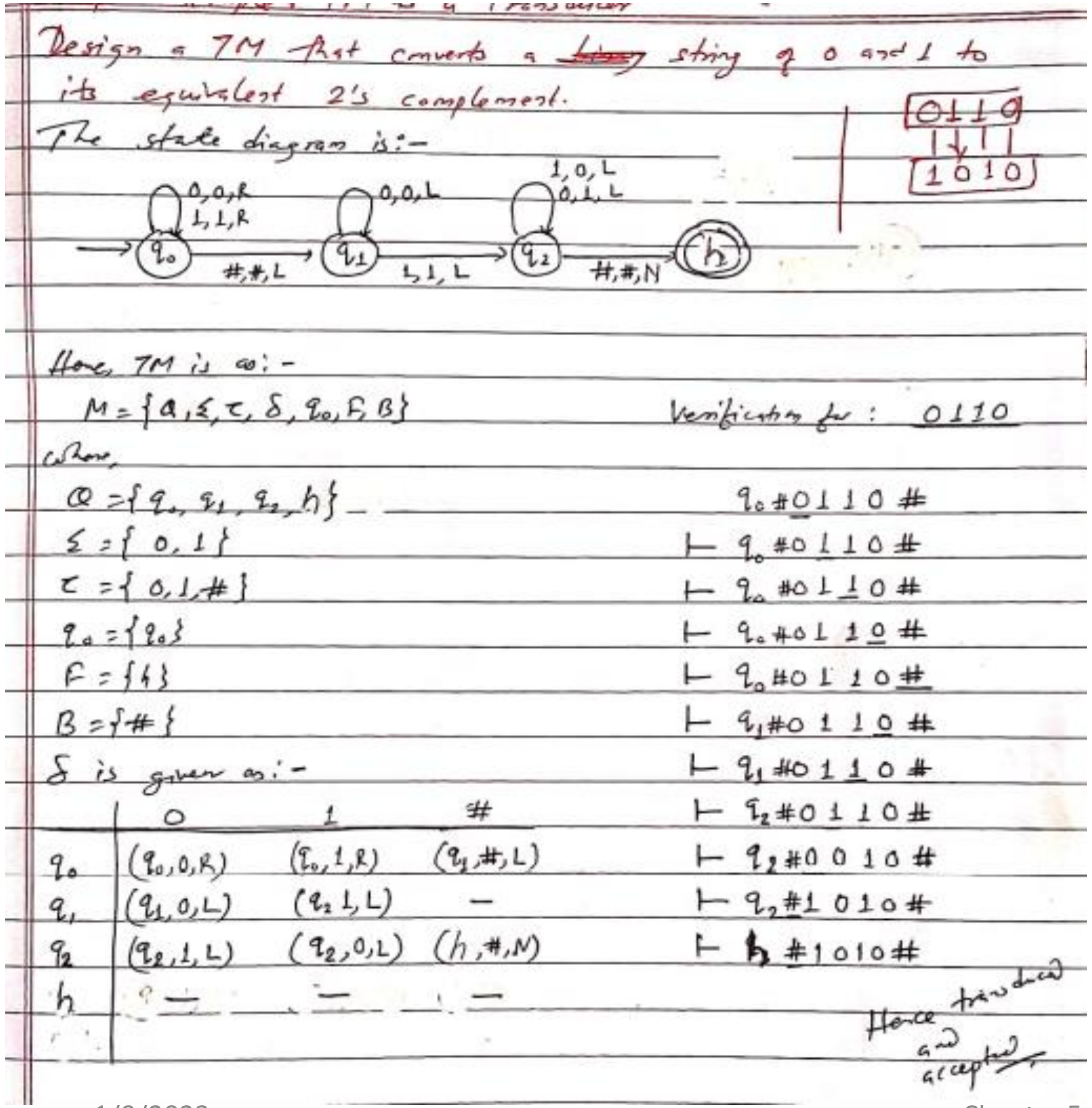
δ is given by the following transition table:-

	0	1	#
q_0	–	–	(q_2, #, R)
q_1	(q_2, 0, R)	(q_1, #, R)	(h, #, N)
q_2	(q_2, 0, L)	(q_1, ~~#~~1, R)	(h, #, N)
h	–	–	–

The transition diagram for above TM will be as shown below, where we assume that q_0 is the initial state and h is a final ~~[illegible]~~ state.



5.13 Numerical: TM as Transducer



0,0,R
1,1,R
q_0
#,#,L
0,0,L
q_1
1,1,L
1,0,L
0,1,L
q_2
#,#,N
h
0110
1010

Design a TM that converts a ~~binary~~ string of 0 and 1 to its equivalent 2's complement.

The state diagram is:-

Here, TM is as:-

$M = \{Q, \Sigma, \Gamma, \delta, q_0, F, B\}$

where,

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

$\Sigma = \{0, 1\}$

$\Gamma = \{0, 1, \#\}$

$q_0 = \{q_0\}$

$F = \{h\}$

$B = \{\#\}$

δ is given as:-

	0	1	#
q_0	$(q_0, 0, R)$	$(q_0, 1, R)$	$(q_1, \#, L)$
q_1	$(q_1, 0, L)$	$(q_2, 1, L)$	—
q_2	$(q_2, 1, L)$	$(q_2, 0, L)$	$(h, \#, N)$
h	—	—	—

Verification for : 0110

$q_0 \#\underline{0}110\#$

$\vdash q_0 \#0\underline{1}10\#$

$\vdash q_0 \#01\underline{1}0\#$

$\vdash q_0 \#011\underline{0}\#$

$\vdash q_0 \#0110\underline{\#}$

$\vdash q_1 \#011\underline{0}\#$

$\vdash q_1 \#01\underline{1}0\#$

$\vdash q_2 \#0\underline{1}10\#$

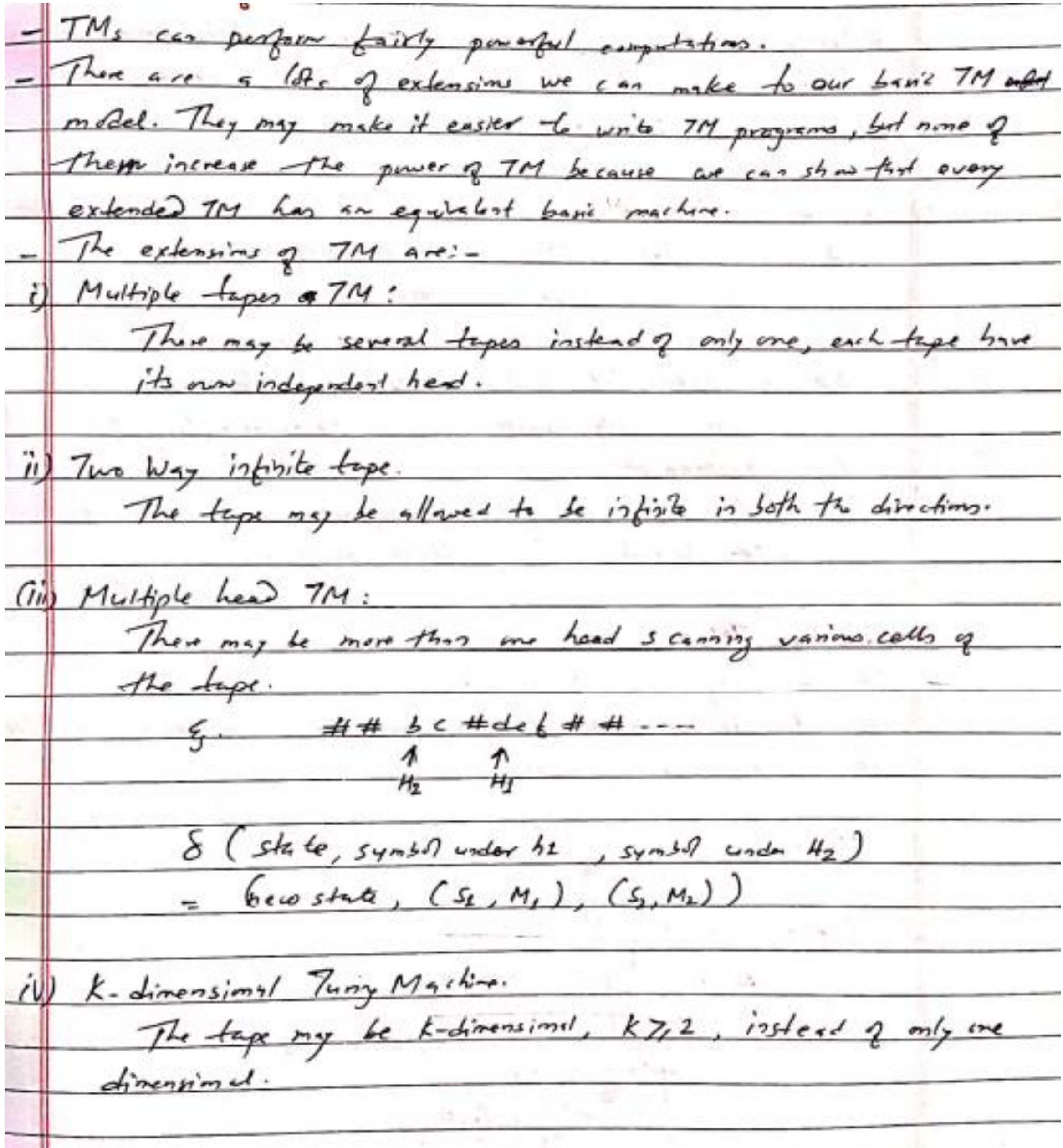
$\vdash q_2 \#\underline{0}010\#$

$\vdash q_2 \underline{\#}1010\#$

$\vdash h \underline{\#}1010\#$

Hence transduced and accepted.

5.14 Extensions of TM



- TMs can perform fairly powerful computations.
- There are a lots of extensions we can make to our basic TM ~~and~~ model. They may make it easier to write TM programs, but none of these increase the power of TM because we can show that every extended TM has an equivalent basic machine.
- The extensions of TM are:-

i) Multiple tapes TM:

There may be several tapes instead of only one, each tape have its own independent head.

ii) Two Way infinite tape.

The tape may be allowed to be infinite in both the directions.

(iii) Multiple head TM:

There may be more than one head scanning various cells of the tape.

Eg. ## b c #def # # ---

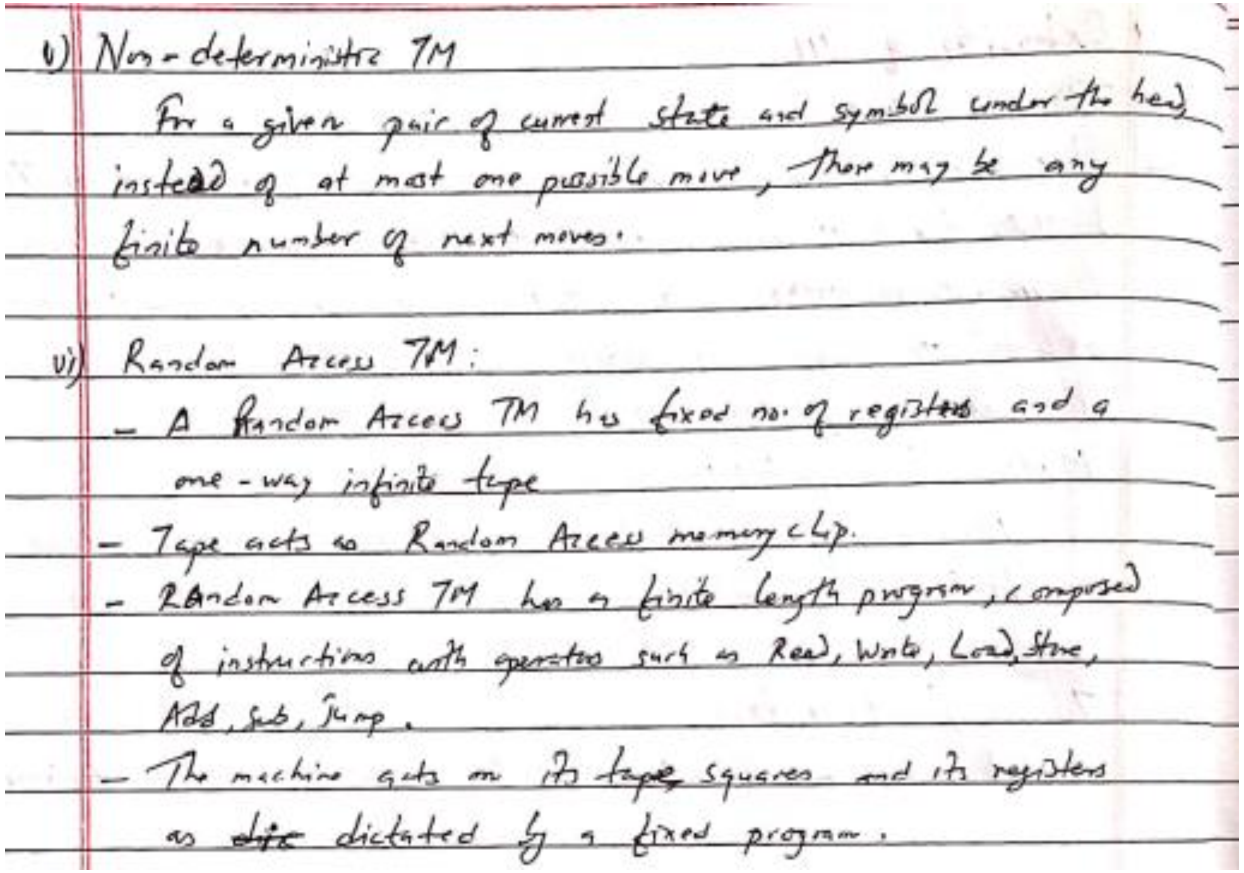
$\uparrow$ H_2 $\uparrow$ H_1

δ (state, symbol under h1 , symbol under H_2)

= (new state, (S_1, M_1), (S_2, M_2))

iv) K-dimensional Turing Machine.

The tape may be K-dimensional, $K \geq 2$, instead of only one dimensional.



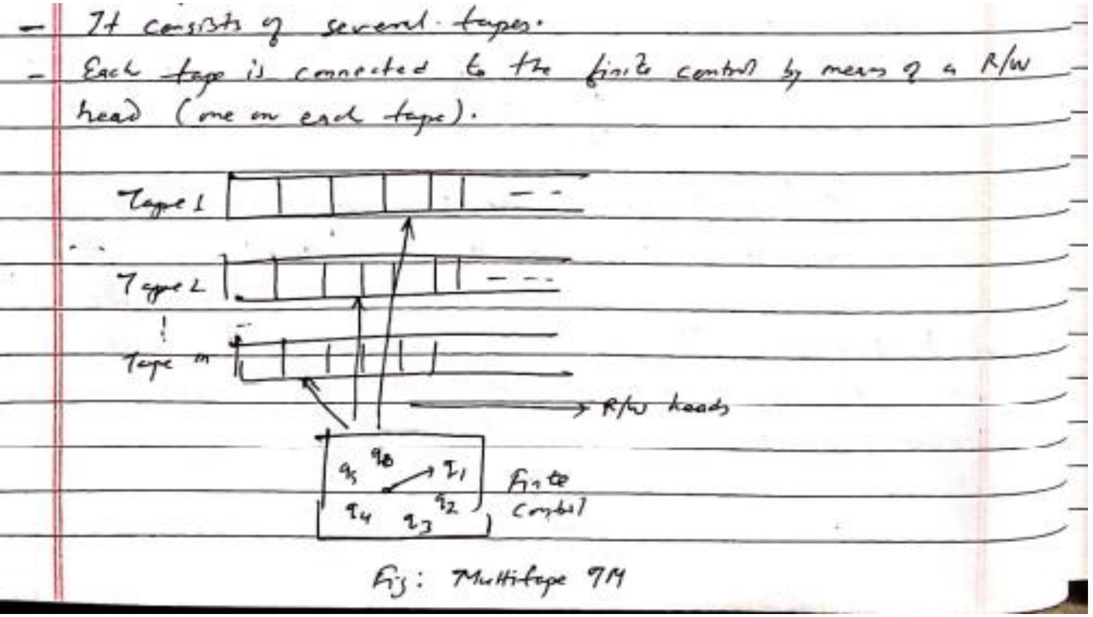
v) Non-deterministic TM

For a given pair of current state and symbol under the head instead of at most one possible move, there may be any finite number of next moves.

vi) Random Access TM:

- A Random Access TM has fixed no. of registers and a one-way infinite tape
- Tape acts as Random Access memory chip.
- RAndom Access TM has a finite length program, composed of instructions with operators such as Read, Write, Load, Store, Add, Sub, Jump.
- The machine acts on its tape squares and its registers as ~~dic~~ dictated by a fixed program.

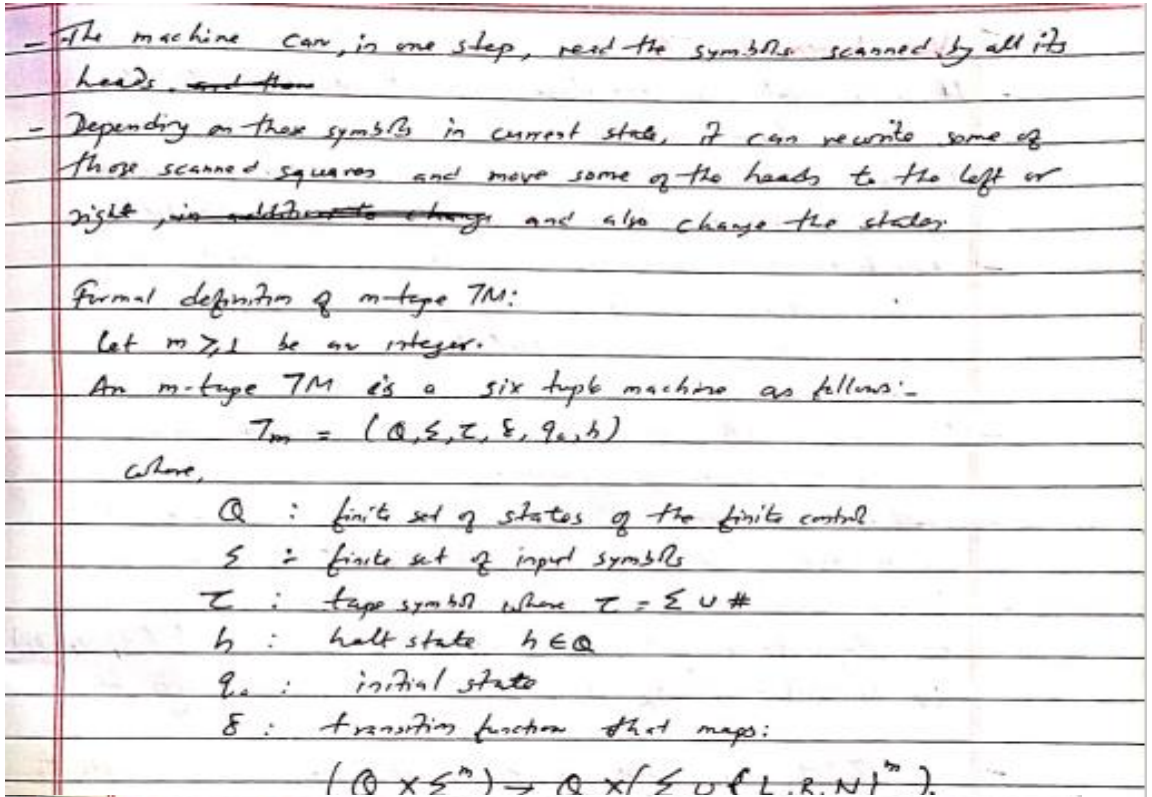
5.14.1 Multi-tape Turing Machine



- It consists of several tapes.
- Each tape is connected to the finite control by means of a R/W head (one on each tape).

Tape 1
Tape 2
Tape m
R/W heads
q_5
q_0
q_1
q_2
q_3
q_4
Finite Control

Fig: Multitape TM



- The machine can, in one step, read the symbols scanned by all its heads. ~~and then~~
- Depending on those symbols in current state, it can rewrite some of those scanned squares and move some of the heads to the left or right, ~~in addition to change~~ and also change the states.

Formal definition of m-tape TM:

Let $m \geq 1$ be an integer.

An m-tape TM is a six tuple machine as follows:-

$$T_m = (Q, \Sigma, \tau, \delta, q_0, h)$$

where,

Q : finite set of states of the finite control

Σ : finite set of input symbols

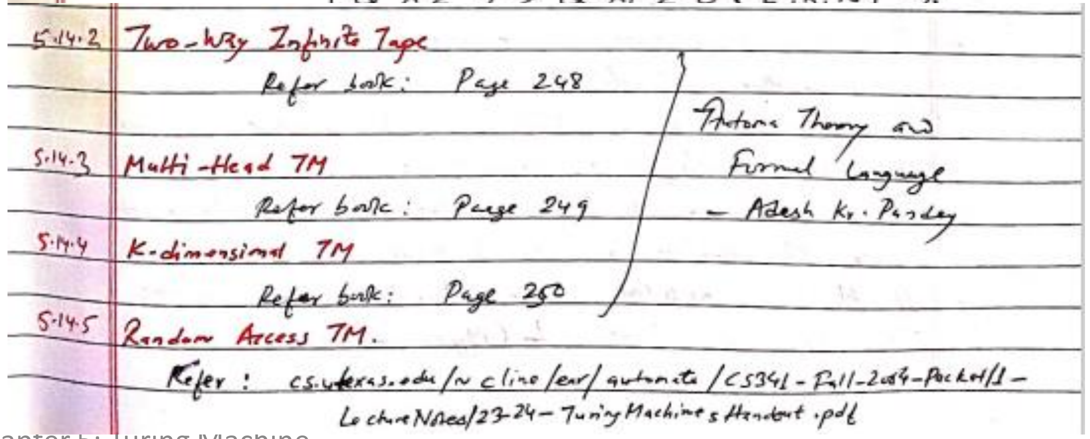
τ : tape symbol where $\tau = \Sigma \cup \#$

h : halt state $h \in Q$

q_0 : initial state

δ : transition function that maps:

$$(Q \times \Sigma^m) \rightarrow Q \times (\Sigma \cup \{L.R.N\}^m).$$



5.14.2 Two-Way Infinite Tape

Refer book: Page 248

5.14.3 Multi-Head TM

Refer book: Page 249

5.14.4 K-dimensional TM

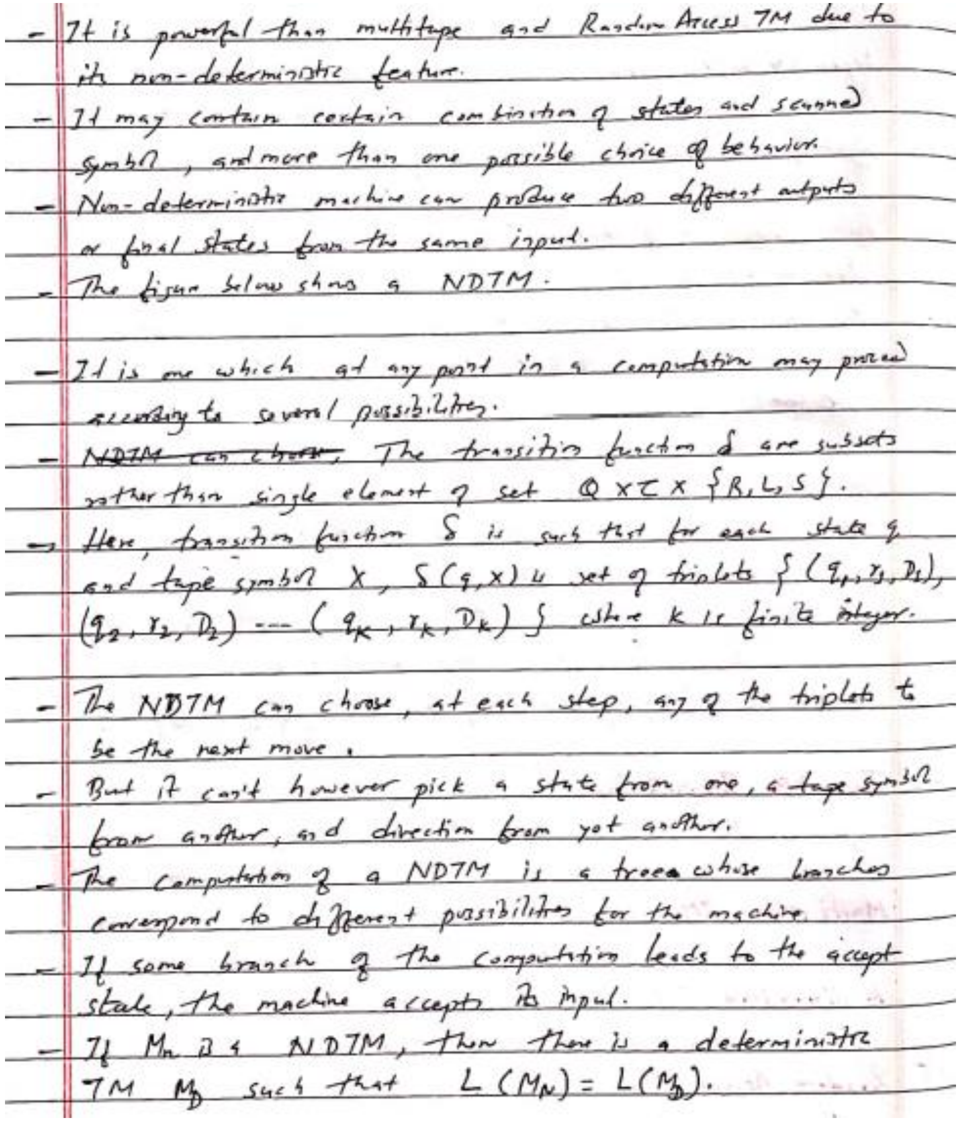
Refer book: Page 250

Automata Theory and Formal Language – Adesh Kr. Pandey

5.14.5 Random Access TM.

Refer : cs.utexas.edu/~cline/ear/automata/CS341-Fall-2004-Packet/1-LectureNotes/23-24-TuringMachinesHandout.pdf

5.15 Non-Deterministic TM

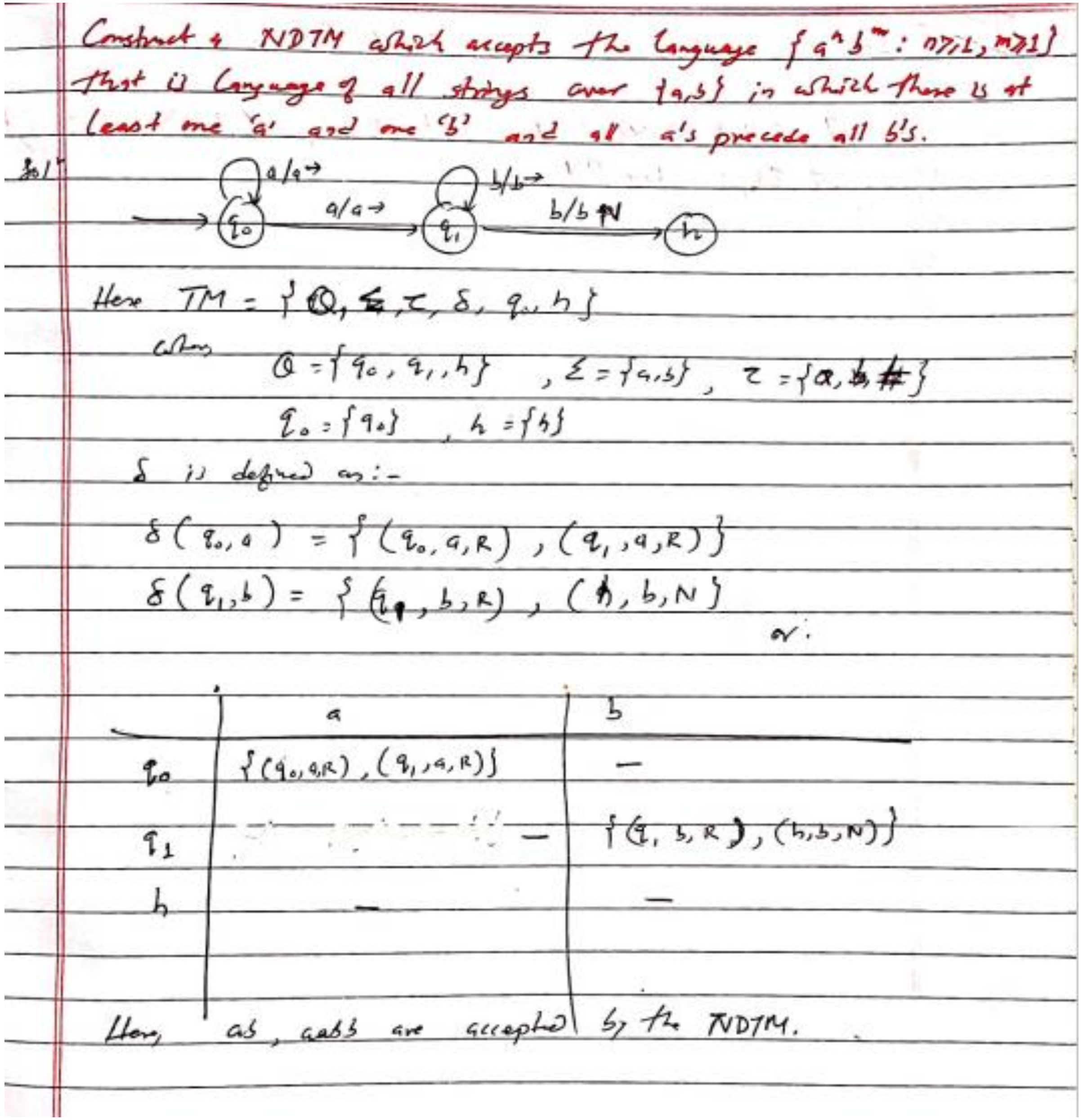


- It is powerful than multitape and Random Access TM due to its non-deterministic feature.
- It may contain certain combination of states and scanned symbol, and more than one possible choice of behavior.
- Non-deterministic machine can produce two different outputs or final states from the same input.
- The figure below shows a NDTM.

- It is one which at any point in a computation may proceed according to several possibilities.
- ~~NDTM can choose,~~ The transition function δ are subsets rather than single element of set $Q \times \Gamma \times \{R, L, S\}$.

→ Here, transition function δ is such that for each state q and tape symbol X, $\delta(q, X)$ is set of triplets $\{(q_1, Y_1, D_1), (q_2, Y_2, D_2) --- (q_K, Y_K, D_K)\}$ where k is finite integer.

- The NDTM can choose, at each step, any of the triplets to be the next move.
- But it can't however pick a state from one, a tape symbol from another, and direction from yet another.
- The computation of a NDTM is a tree whose branches correspond to different possibilities for the machine.
- If some branch of the computation leads to the accept state, the machine accepts its input.
- If M_N is a NDTM, then there is a deterministic TM M_D such that $L(M_N) = L(M_D)$.



Construct a NDTM which accepts the language $\{a^n b^m : n \geq 1, m \geq 1\}$ That is Language of all strings over $\{a,b\}$ in which there is at least one 'a' and one 'b' and all a's precede all b's.

Sol^n:

a/a→
b/b→
a/a→
b/b N
q_0
q_1
h

Here $TM = \{Q, \Sigma, \Gamma, \delta, q_0, h\}$

where $Q = \{q_0, q_1, h\}$, $\Sigma = \{a,b\}$, $\Gamma = \{a, b, \#\}$

$q_0 = \{q_0\}$, $h = \{h\}$

δ is defined as:-

$\delta(q_0, a) = \{(q_0, a, R), (q_1, a, R)\}$

$\delta(q_1, b) = \{(q_1, b, R), (h, b, N\}$

or:

	a	b
q_0	$\{(q_0,a,R), (q_1,a,R)\}$	–
q_1	–	$\{(q_1, b, R), (h,b,N)\}$
h	–	–

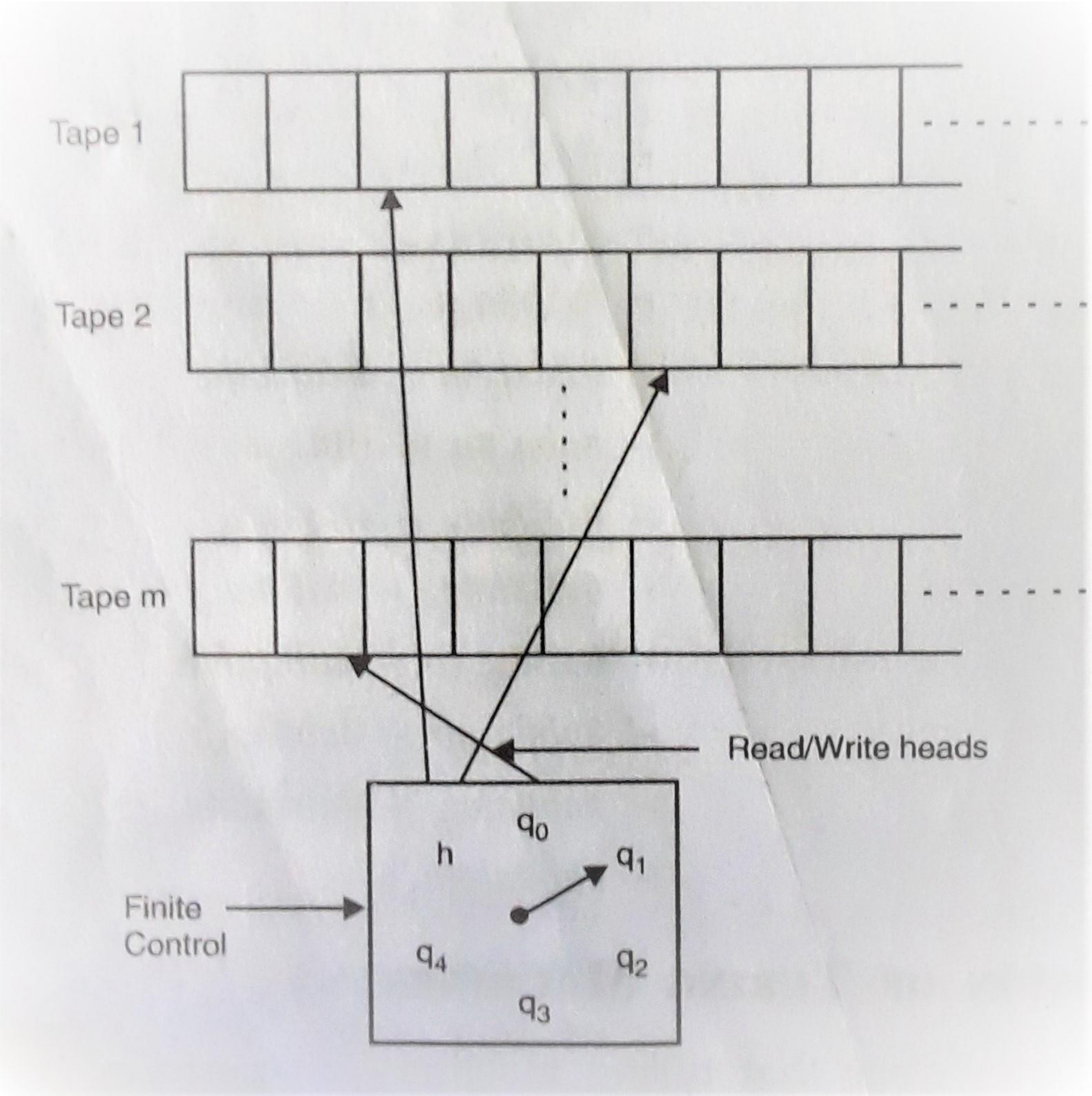
Here, ab, aabb are accepted by the NDTM.

5.14 Extensions of TM

- TMs can perform fairly powerful computations
- There are a lot of extensions we can make to our basic TM model.
- They may make it easier to write TM programs,
- But none of them increase the power of TM because we can show that every extended TM has an equivalent basis machine.
- The extensions of TM are:
  1. Multiple tapes TM
  2. Two-way Infinite tape TM
  3. Multiple head TM
  4. K-dimensional TM
  5. Non-deterministic TM
  6. Random Access TM

1. Multiple tapes TM

- There may be several tapes instead of one
- Each tape have its own independent head.
- Each tape is connected to the finite control by means of a R/W head (one in each tape).

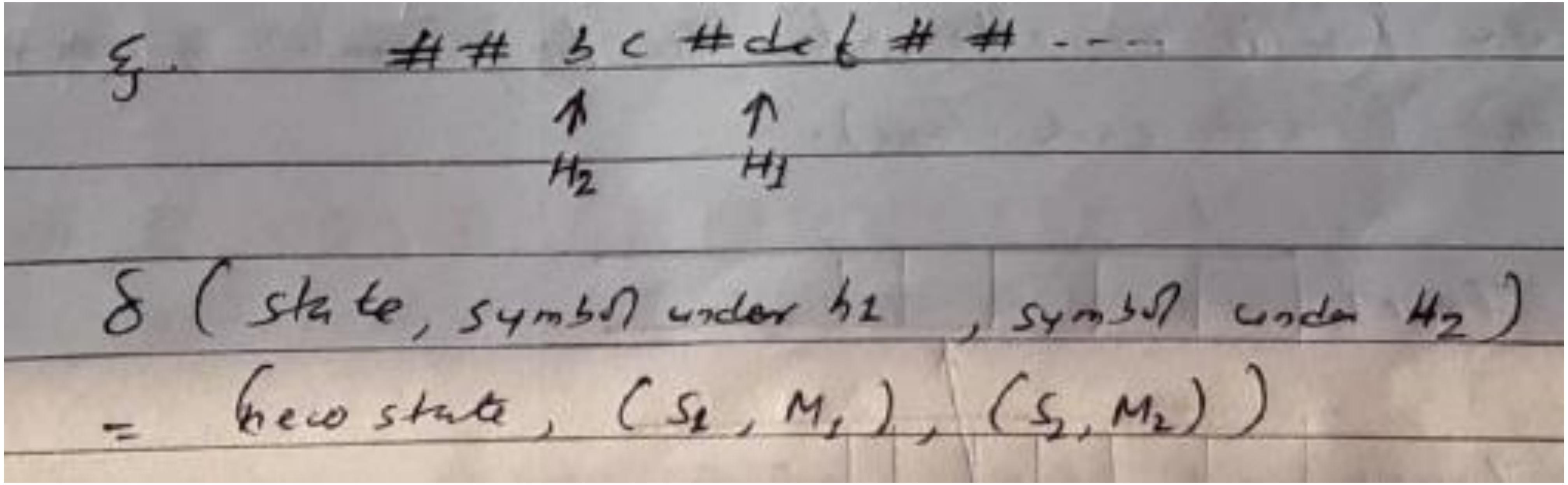


2. Two way infinite tape

- The tape may be allowed to be infinite in both the directions.

3. Multiple Head TM

There may be more than one head scanning carious cells of the tape.



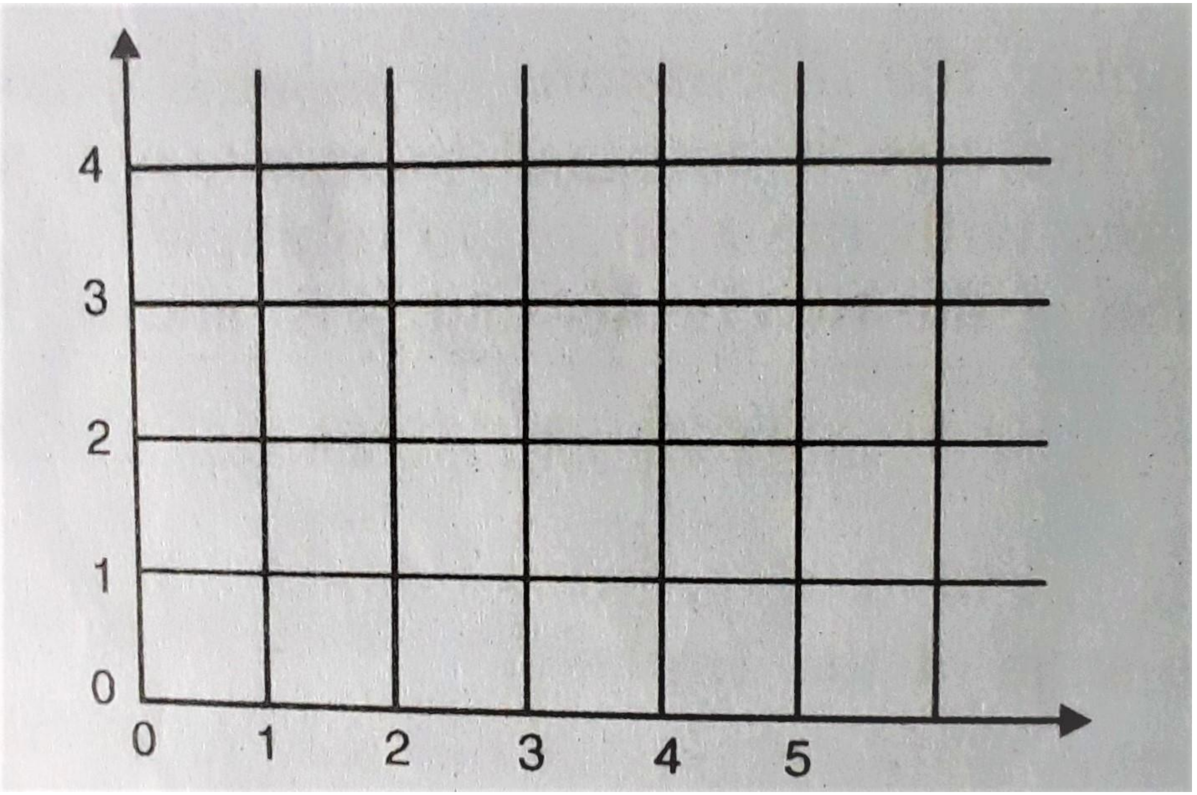
Eg. ## b c # d e f # # ---

δ (state, symbol under h1 , symbol under H_2)

= (new state, (S_1 , M_1), (S_2 , M_2))

4. K-dimensional TM

- The tape may be K-dimensional
- K>=2 instead of only one dimension.

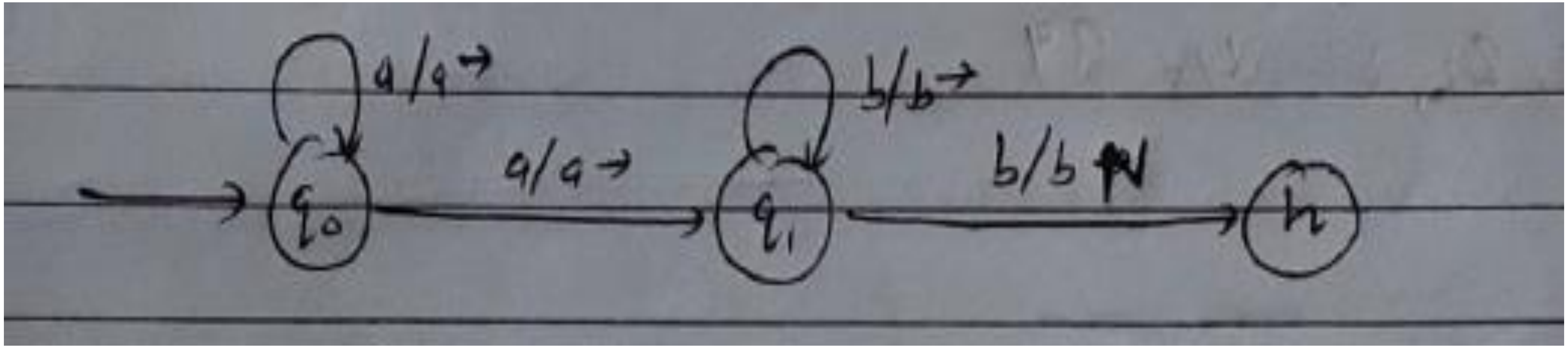


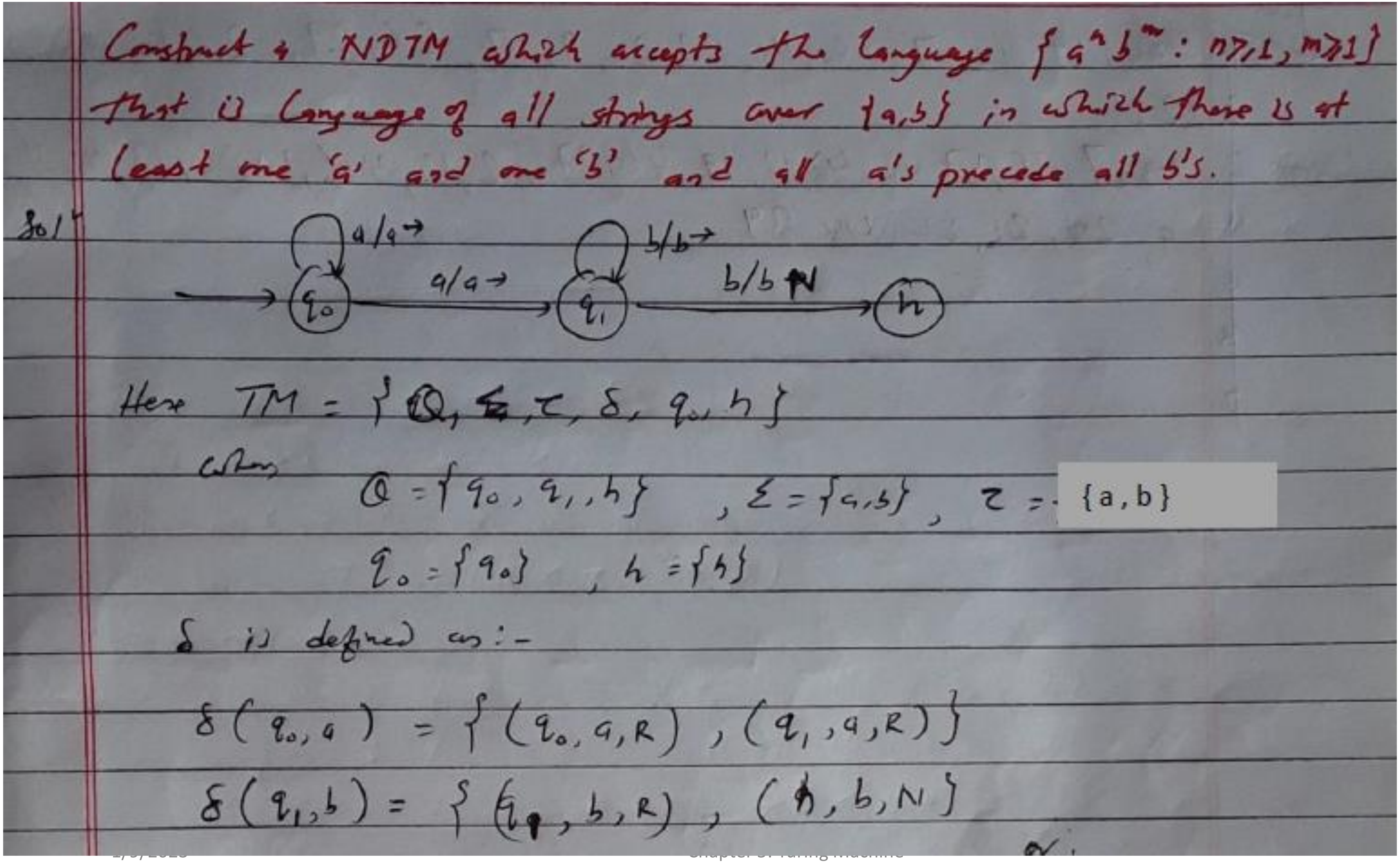
5. Non-deterministic TM

For a given pair of current state and symbol under the head, instead of at most one possible move, there may be any finite number of next moves.

More powerful than multi-tape TM and Random access TM due to its non-deterministic feature.

Figure below shows a NDTM.




a/a→
a/a→
b/b→
b/b N
q_0
q_1
h

Construct a NDTM which accepts the language $\{a^n b^m : n \geq 1, m \geq 1\}$ that is language of all strings over $\{a,b\}$ in which there is at least one 'a' and one 'b' and all a's precede all b's.

Sol^n

Here $TM = \{Q, \Sigma, \tau, \delta, q_0, h\}$

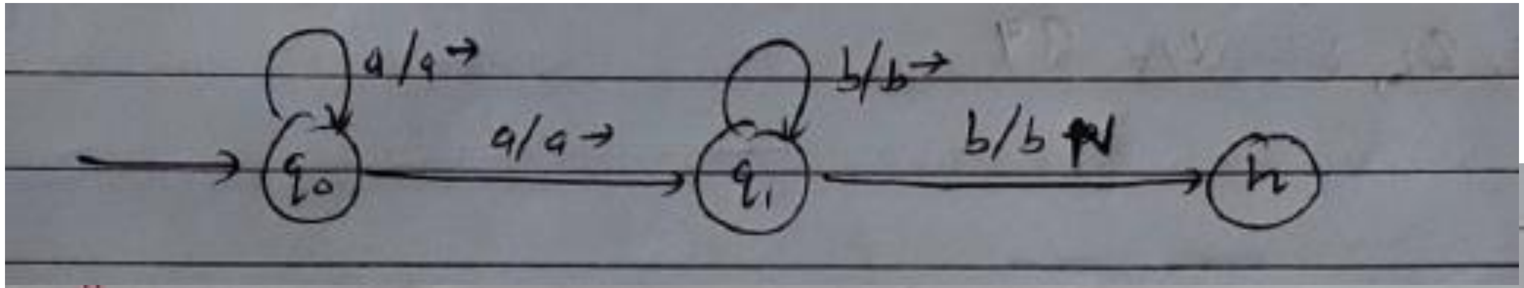
where $Q = \{q_0, q_1, h\}$, $\Sigma = \{a,b\}$, $\tau =$ {a, b}

$q_0 = \{q_0\}$, $h = \{h\}$

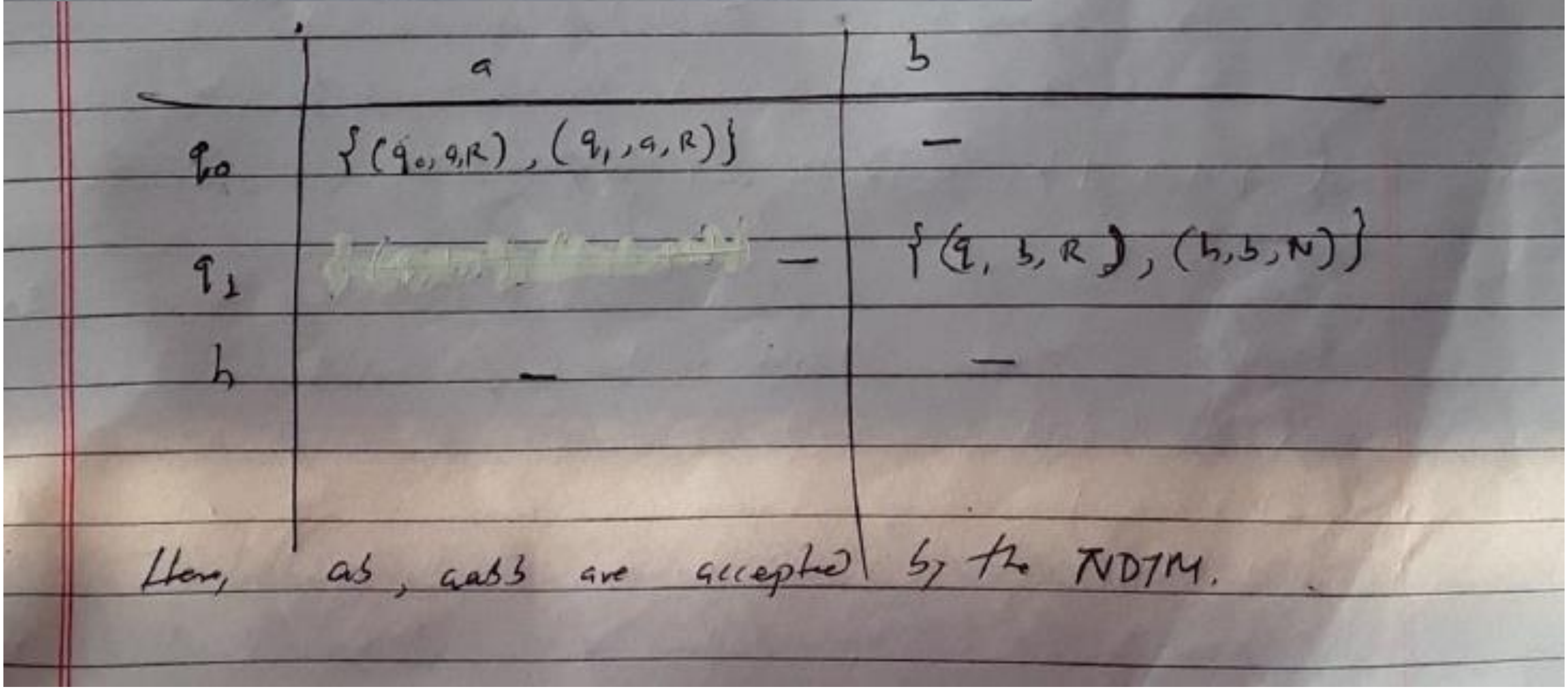
δ is defined as:-

$\delta(q_0, a) = \{(q_0, a, R), (q_1, a, R)\}$

$\delta(q_1, b) = \{(q_1, b, R), (h, b, N\}$



or.



	a	b
q_0	$\{(q_0, a, R), (q_1, a, R)\}$	—
q_1	—	$\{(q, b, R), (h, b, N)\}$
h	—	—

Here, ab, aabb are accepted by the NDTM.

6. Random Access TM

- A Random Access TM has fixed no. of registers and a one-way infinite tape.
- Tape acts as Random Access Memory chip.
- Random Access TM has a finite length program, composed of instructions with operators such as READ, WRITE, STORE, ADD, SUB, JUMP.
- The machine acts on its tape squares and its registers as dictated by a fixed program.

References:

- https://www.cs.utexas.edu/~cline/ear/automata/CS341-Fall-2004-Packet/1-LectureNotes/23-24-TuringMachinesHandout.pdf
- Pandey A.K., An Introduction to AutomataTheory and Formal Languages, S.K. Kataria & sons, pp. 245-250

End of chapter