type-1 Context sensitive

$G = (N, T, P, S)$

$\alpha \longrightarrow \beta$ where $\alpha, \beta \in (N \cup T)^*$

$|\alpha| \leq |\beta|$ Linear ~~sensitive~~ bounded automation.

→ DCFL's are not closed under union

Turing machine.

1) The general model of morden computer.
2) Turing machine introduced by Alan turing 1936.

DFA / PDA. turing machine

1) Finite state and alphabet ✓
2) transition function ✓

Differences

(1) It may not stop.

A turing machine is a 5-tuple $(K, \Sigma, \delta, S, H)$ where,

(1) K is a ~~set~~ finite set of states

2) Σ is a finite alphabet combining blank symbol "#" and left-end symbol $\triangleright$

3) $H \subset K$ called the set of halting states.

4) $\delta : (K-H) \times \Sigma \longrightarrow K \times (\Sigma \cup \{L \cup R\})$ such that

(1) for $q \in K-H$, is $\delta(q, \triangleright) = (p, b)$ then $b = R$.

(2) $q \in K-H$, $a \in \Sigma$ is $\delta(q, a) = (p, b)$ then $b \neq \triangleright$

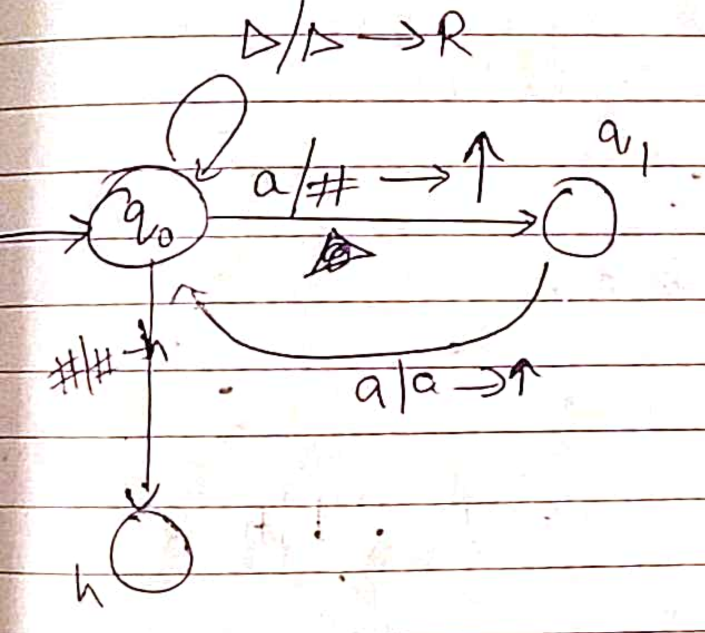
$M = (\cancel{K}, \Sigma, \delta, S, \{h\})$

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

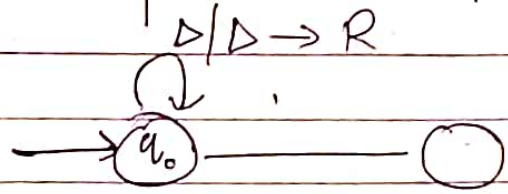
$\Sigma = \{a, \#, \triangle\}$

$S = q_0.$

δ	q	γ	$\delta(q, \gamma)$
	q_0	a	$(q_1, \#)$
	q_0	$\#$	$(h, \#)$
	q_0	$\triangleright$	(q_0, R)
	q_1	a	(q_0, a)
	q_1	$\#$	(q_0, R)
	q_1	$\triangleright$	(q_1, R)



δ	q	γ	$\delta(q, \gamma)$
	q_0	a	(q_0, L)
	q_0	$\#$	$(h, \#)$
	q_0	$\triangleright$	(q_0, R)

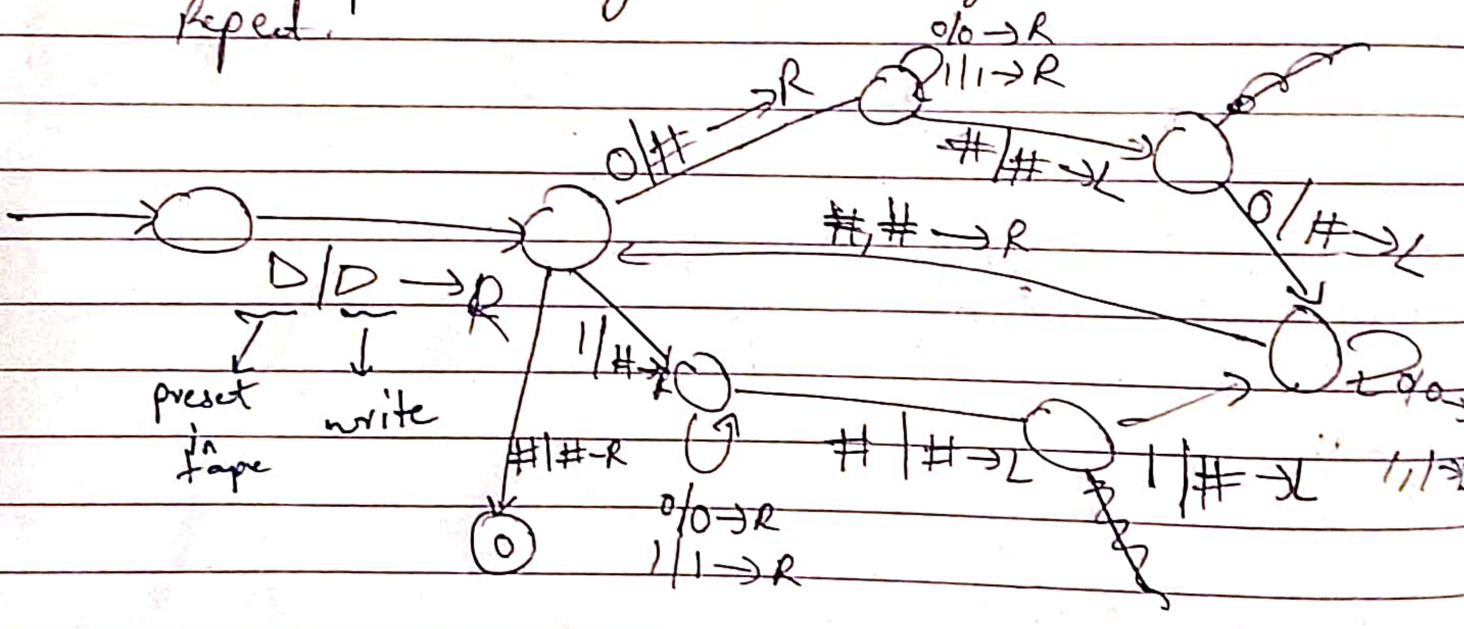


Build a turing machine :

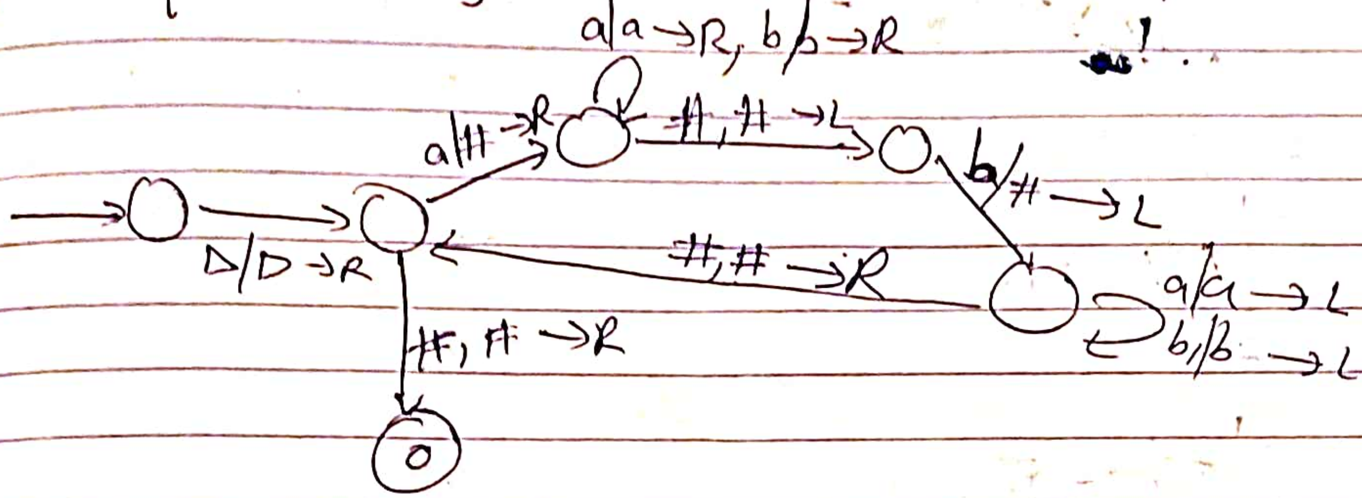
$L = \{ w \in (0+1)^* \mid w = w^R \}$

D	a	b	c	d	#	#

(1) See a symbol, print #
(2) Move right until see a #
(3) Take a left move, see the symbol
(4) if same symbol seen then replace it by the # move left until #
(5) Replace the symbol, take a right move.
Repeat.



Eg:- $L = \{a^n b^n \mid n \geq 0\}$



Configuration of a turing machine:-

Configuration of a turing machine Consists of the following Components:-

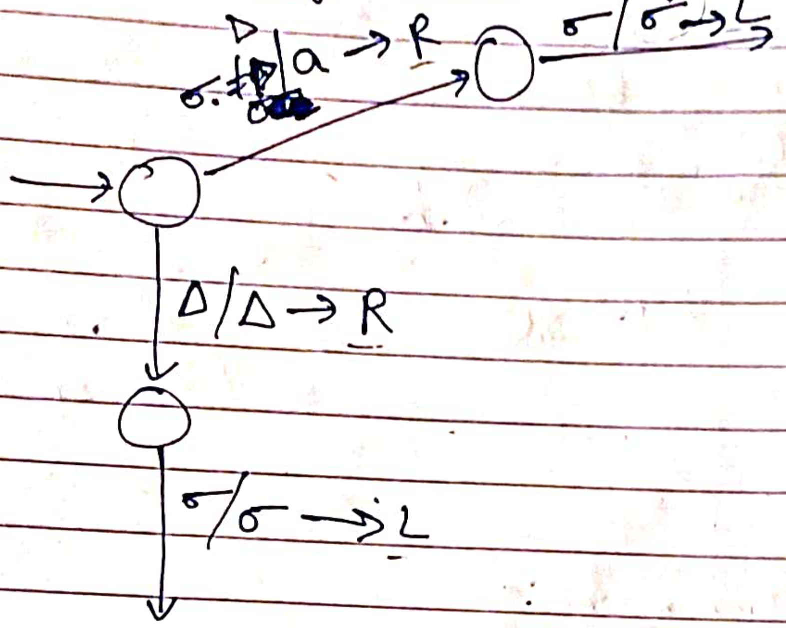
(1) The current state
(2) The Current tape content
(3) current head position

Combination of turing Machine:-

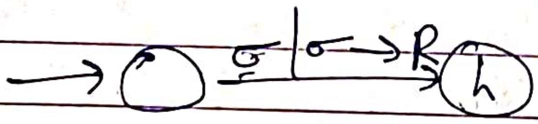
Step1:- Define some Simple mechanism.

Step2:- combine them to make more Complex machines.

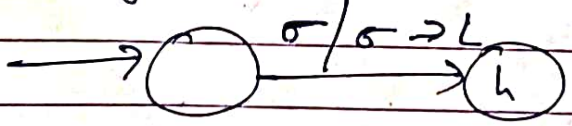
Symbol writting machine :— M_a (or) a

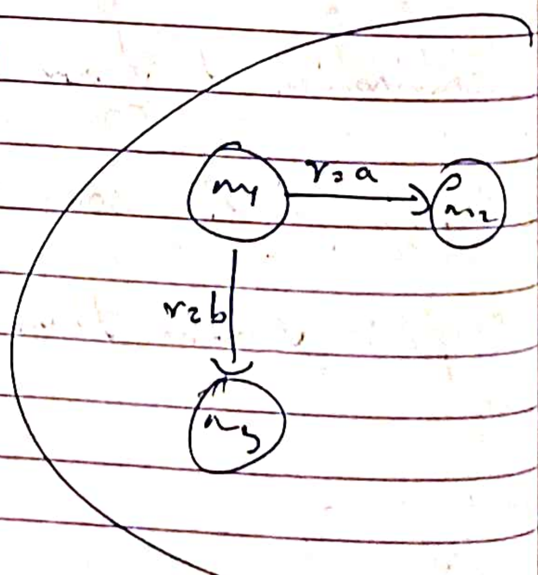


Head moving machine :— M_R (or) R.



Similarly M_L :—





Q (c) M where the head moves two position two right and hault

Ans) $>RR$.

Q TM that moves the head Position to the first blank symbol in the right.

Ans) $\rhd R \circlearrowleft r \neq \#$

$R_{\#}$ $\quad$ $L_{\#}$

$> L \circlearrowleft r \neq \#$

Q Copy machine

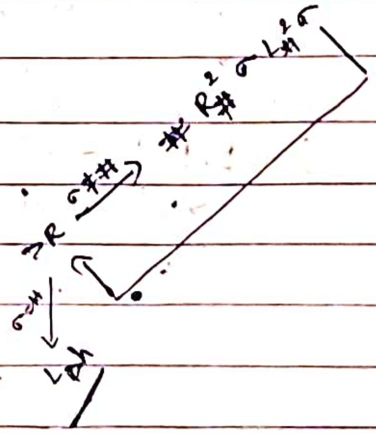
▷	w	#	# . . .

▷	w	#	w	#	#	. . .

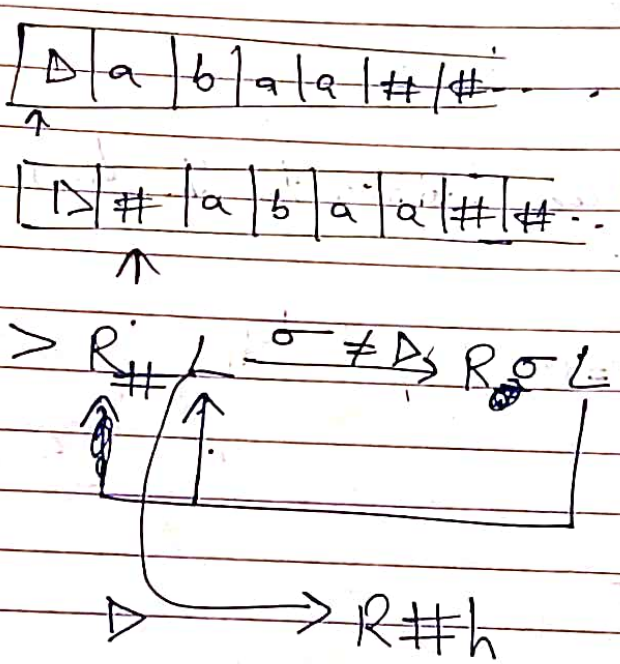
~~$>R \xrightarrow{\sigma \neq \#} R^2_{\#}$~~

$>R \xrightarrow{\sigma \neq \#} \#\, R^2_{\#}\, \sigma\, L^2_{\#}\, \sigma$

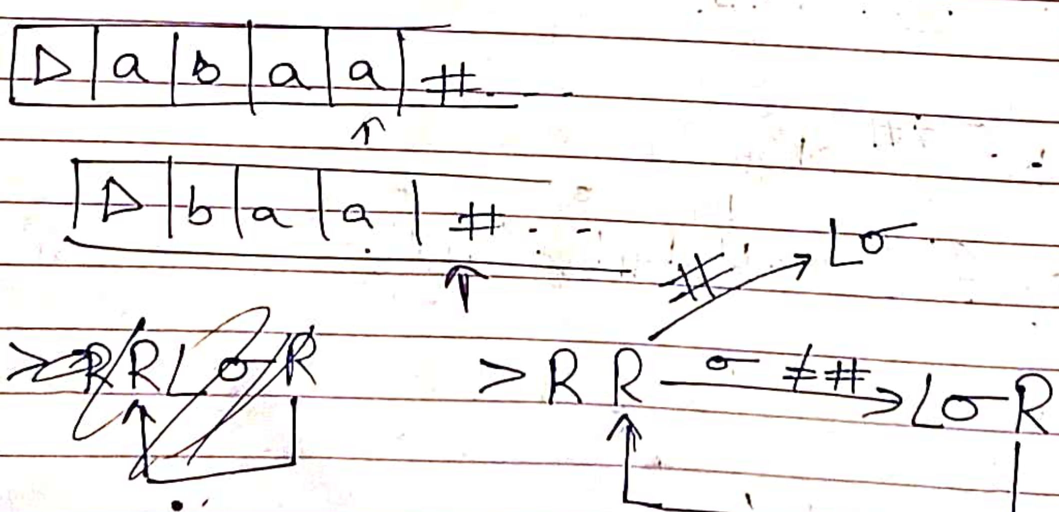
$\sigma = \# \longrightarrow L_{\rhd} h$



② Right shift machine :-



③ Left shift machine :-



Date ______
Page ______

Turning machine as a language acceptor :-

Let $M = (K, \Sigma, \delta, S, H)$ be a TM such that

$H = \{h_a, h_r\}$ consits of two halting state, such that

(1) any halting configuration whose state component is h_a is called accepting configuration.

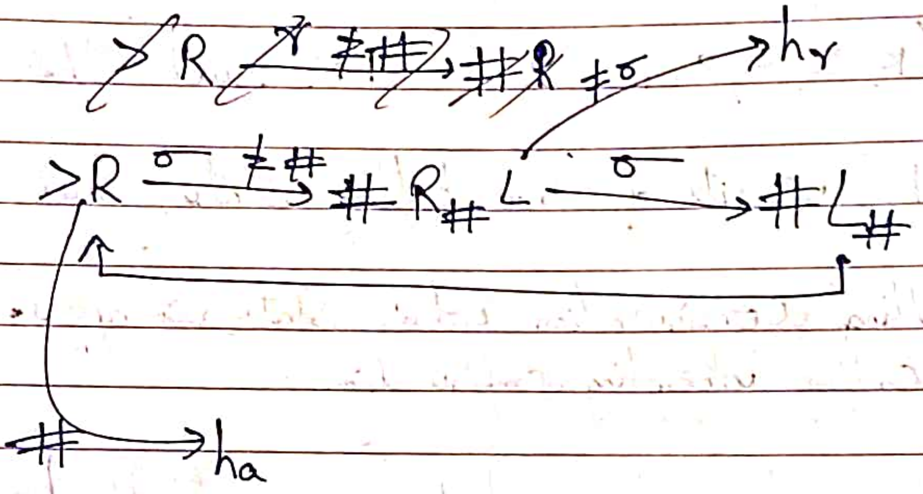
(2) Any halting Config. whose stat. Component if h_r is called rejecting Configuration.

We say M accepts $w \in \{\Sigma - \{\#, \Delta\}^*\}$ if $(S, \Delta \# w)$ yeilds an accepting configuration.

(1) If $w \in L$, M accepts it
(2) if $w \notin L$, M recjects it.

A language is called recursive if $\exists$ a TM M that decides that.

$L = \{ ww^R \mid w \in \Sigma^* \}$

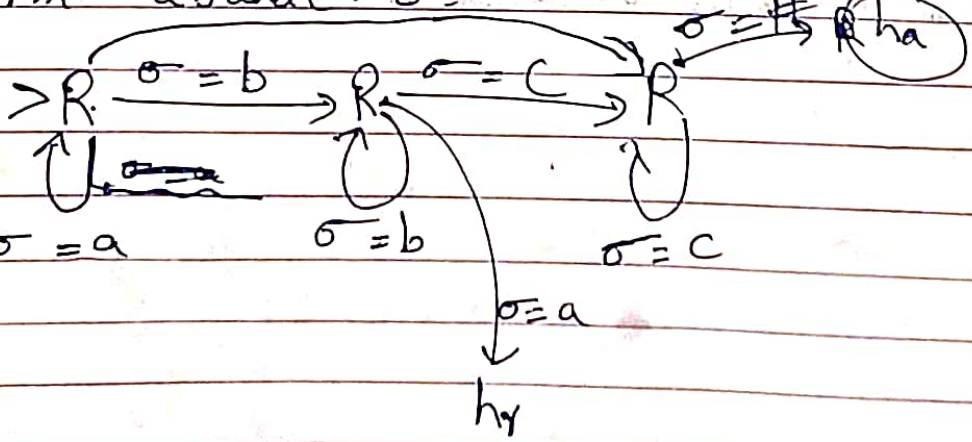


1) $L = \{ a^n b^n c^n \mid n \geq 0 \}$

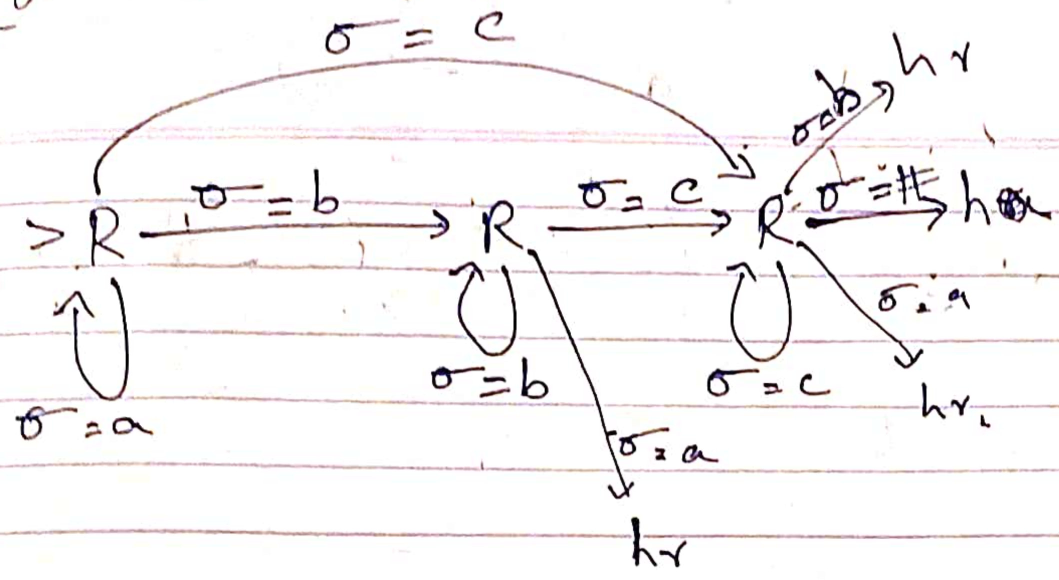
$L = \{ ww \mid w \in (a+b)^* \}$

1) machine to check whether the given string is of the form a b and c.

Sol form $a\ b$ and c. $\sigma = c$



aabcb.



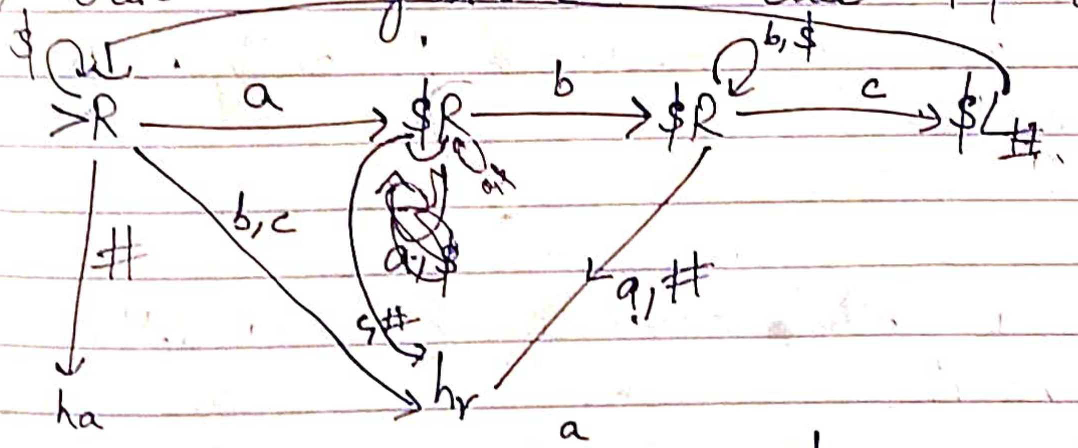
Turing Machine as a language acceptor

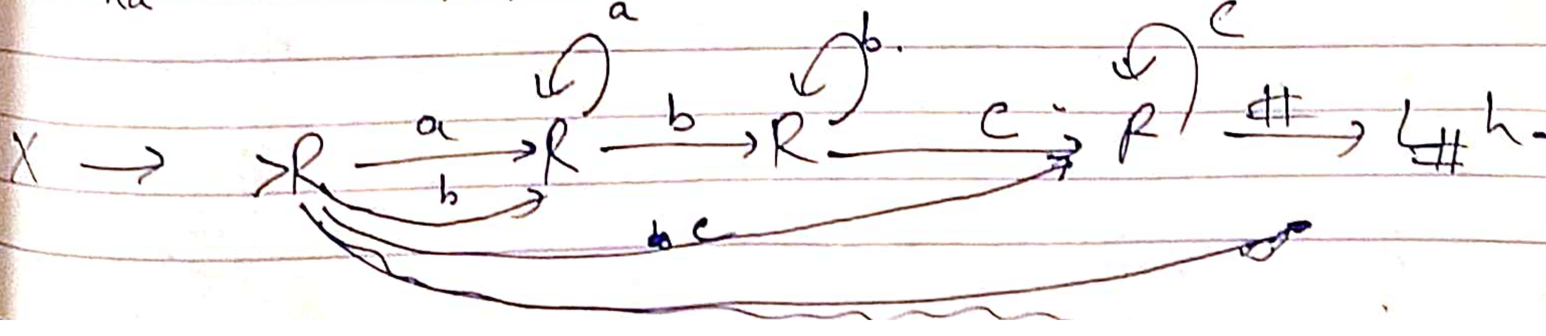
$L = \{a^n b^n c^n \mid n \geq 0\}$

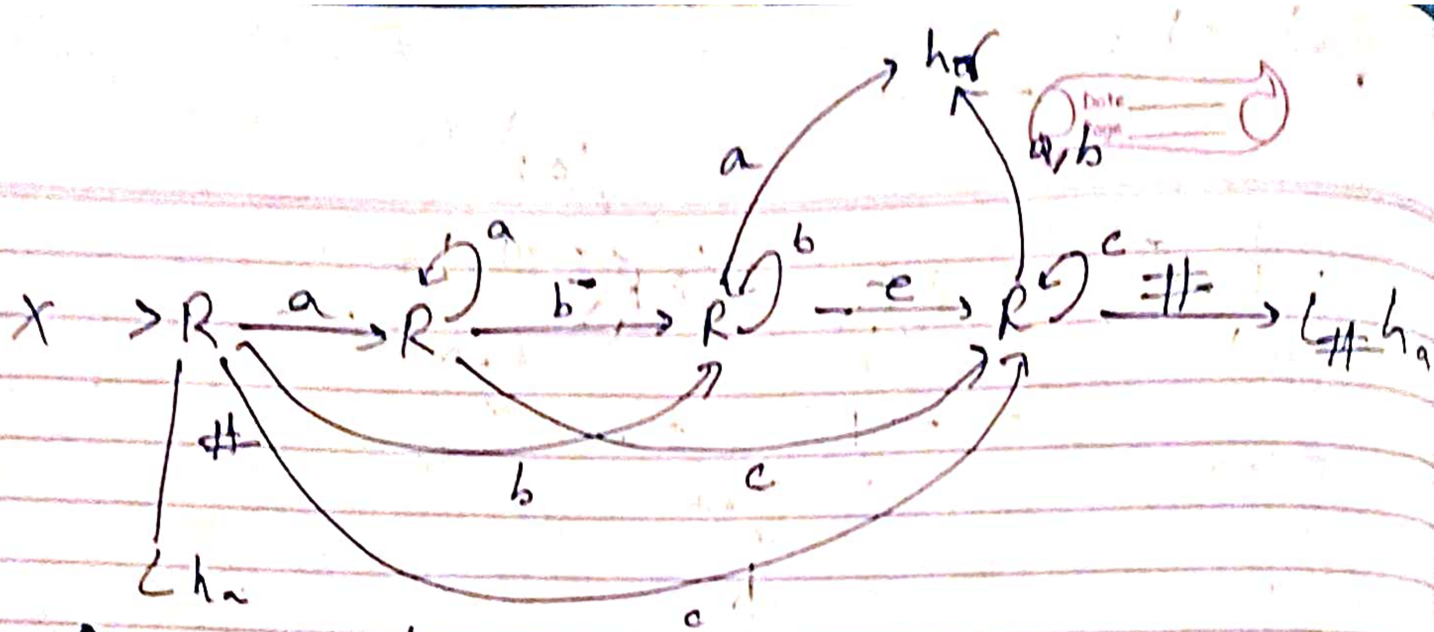
The idea :-

(1) First build a turning Machine that checks whether the string is in the correct form (X).

(2) buid a turing machine that checks $|w|_a = |w|_b = |w|_c$.







Ans $> XY$

<u>Turing Machine as a Computing device</u>

Let $M = (K, \Sigma, \delta, S, \{h\})$ be a turing machine. Let $\Sigma_0 = \Sigma - \{\#, \Delta\}$ be an alphabet and $w \in \Sigma_0^*$.

$$(S, \Delta\underline{\#}w\#) \xrightarrow[M]{*} (h, \Delta\underline{\#}y\#) \text{ for some } y \in \Sigma_0^*$$

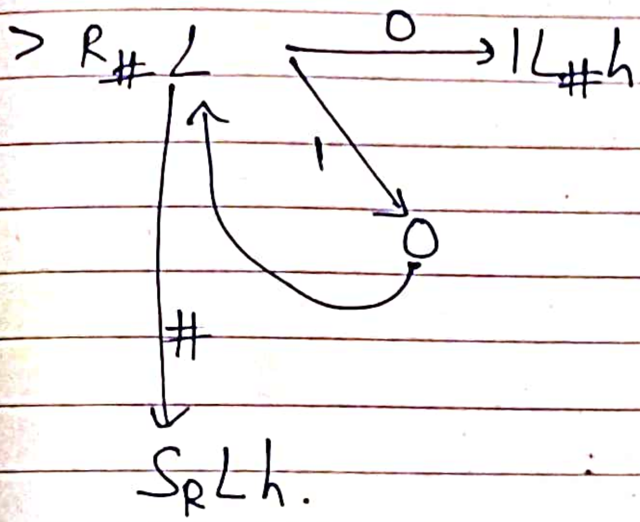
Then y is called the output of M on input w and denoted by M(w)

$f(w) = y$.

A function is called recursive if $\exists$ a T.M that accepts f.

$w \in (0+1)^*$

$f(n) = n+1$



$\triangleright \# 101101 \#$

$\triangleright \# 101110 \#$

$f(w) = ww.$

Eg:- $\triangleright \underline{\#} w \#$

$\downarrow$

$\triangleright \underline{\#} w w \#$

~~$R_{\#}$~~ $> C S_L L_{\triangleright} R$

Recursively enumerable language :-

A language L is called r.e if for every $w \in L$ $\exists$ a turning machine M that halts in w.

Fact Every recursive language is r.e ~~recursive~~.

Fact 2 :- Recursive languages are closed under

(1) intersection
(2) Union
(3) Complement
(4) reversal
(5) Concatination
(6) *

Variations of turning machines :-

(1) Non-deterministic turing machine.

(2) Multi tape turing machine.

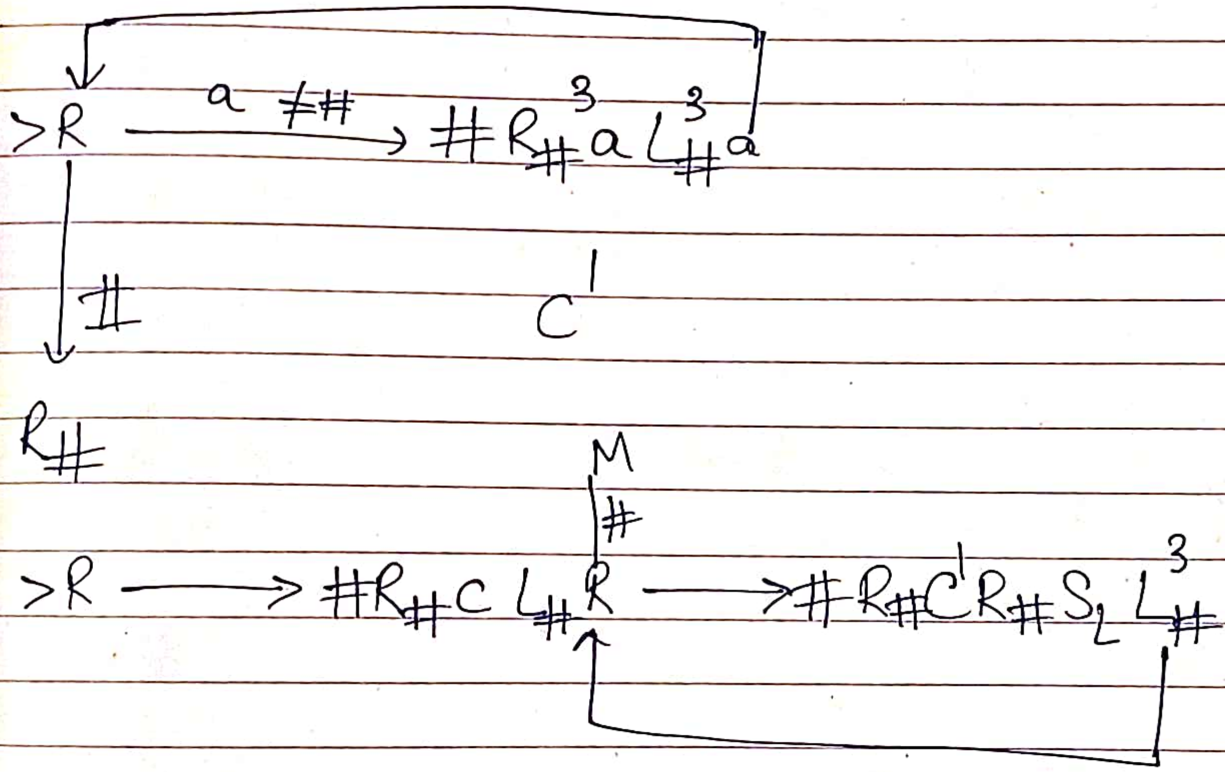
(3) Multi head turing machine.

(4) two way infinite tape turing machine.

$f(m,n) = m+n$ $\qquad\qquad$ $\triangleright\#\ 11111\ \#\ 111\ \#$

$\triangleright\#\ 11111111\ \#$

$f(m \times n) = m \times n.$



$M \quad >R_{\#} L^{\circ} \xrightarrow{\neq \Delta} R S_L$