Construction of Turing m/c

→ There are some basic guidelines for designing Turing m/c.

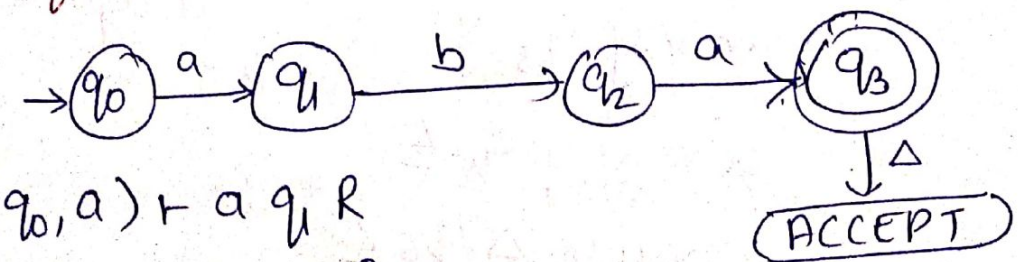
(i) The fundamental objective in scanning a symbol by R/W header is to know what to do in future.

The machine must remember the past symbols scanned. The turing m/c can remember this by going to the next unique state.

(ii) The number of states must be minimized. This can be achieved by changing the state only when there is a change in the written symbol or when there is change in the movement of R/W head.

eg ① Design Turing m/c which accepts the language of "aba" over $\Sigma = \{a, b\}$.

Sol"



$\delta(q_0, a) \vdash a\, q_1\, R$

$\delta(q_1, b) \vdash b\, q_2\, R$

$\delta(q_2, a) \vdash a\, q_3\, R$

$\delta(q_3, \Delta) \vdash$ Accept / Halt

Present state	a	b	Δ
→ q_0	(q_1, a, R)	—	—
q_1	—	(q_2, b, R)	—
q_2	(q_3, a, R)	—	—
(q_3)	—	—	(Halt, Δ, S) stop.

Q2. Construct TM for $L = \{a^n b^n\}$ where $n \geq 1$.

Sol^n Apply following logic :-

As we know number of b (occurrence of b) is equal to occurence of a. So read out 'a' mark it by A & then move ahead along the input tape and findout the b if found then convert it to B. Repeat this process for all a's & b's.

If there is some kind of unequality in no. of a's & b's then TM will not end up in Halt state.

Let consider string of given language. aabb & construct TM.

a a b b Δ

↑

A a b b Δ

↑

A a b b Δ

↑

A a B b Δ

↑

A a B b Δ

↑

A a B b Δ

↑

A A B b Δ

↑

A A B b Δ

↑

A A B B Δ

↑

A A B B Δ

↑

A A B B Δ

↑

A A B B Δ

↑

A A B B Δ

↑

Start with q_0

Convert it to A & move Right in search of B

Skip it, move ahead

Convert it to B & move Left to search A

Move left in searching A.

Without change move Right in searching a.

Convert it to A & move Right in searching b without any changes.

Move Right without change.

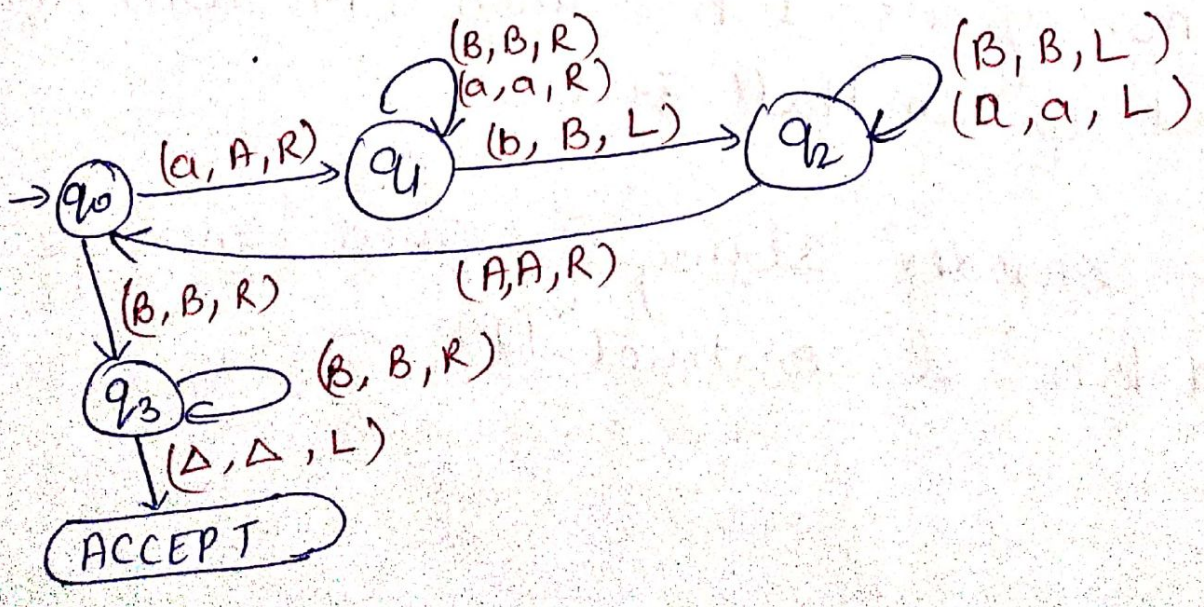
Convert b into B & move left in searching A.

Move left & no change.

Move right in searching a.

Move right in searching Δ & no change.

M/C Halts ∴



Design Turing M/c which recognize the i/p language having a substring as 101 & replace every occurence of 101 by 110.

TM for

Sol" There are 2 cases to generate this language.

Case (i) If just next symbol of pattern ie. 101"0" is zero

Case (ii) If just next symbol of pattern is 1 ie 101|1.

Case(i) During searching if we found 1st symbol of pattern as 1 then write 1 & move in Right in searching of 0.

If 0 is found then write 1 & move Right in searching of 1.

If 1 is found then write 0 & move right.

eg.

$\underline{0}\ 1\ 1\ 0\ 1\ 0\ \Delta$	$q_0\ \underline{0}\ 1\ 1\ 0\ 1\ 0\ \Delta$
$0\ \underline{1}\ 1\ 0\ 1\ 0\ \Delta$	$0\ q_1\ \underline{1}\ 1\ 0\ 1\ 0\ \Delta$
$0\ 1\ \underline{1}\ 0\ 1\ 0\ \Delta$	$0\ 1\ q_1\ \underline{1}\ 0\ 1\ 0\ \Delta$
$0\ 1\ 1\ \underline{0}\ 1\ 0\ \Delta$	$0\ 1\ 1\ q_1\ \underline{0}\ 1\ 0\ \Delta$
$0\ 1\ 1\ 1\ \underline{1}\ 0\ \Delta$	$0\ 1\ 1\ 1\ q_2\ \underline{1}\ 0\ \Delta$
$0\ 1\ 1\ 1\ 0\ \underline{0}\ \Delta$	$0\ 1\ 1\ 1\ q_3\ q_3\ \underline{0}\ \Delta$
$0\ 1\ 1\ 1\ 0\ 0\ \underline{\Delta}$	Halt State.

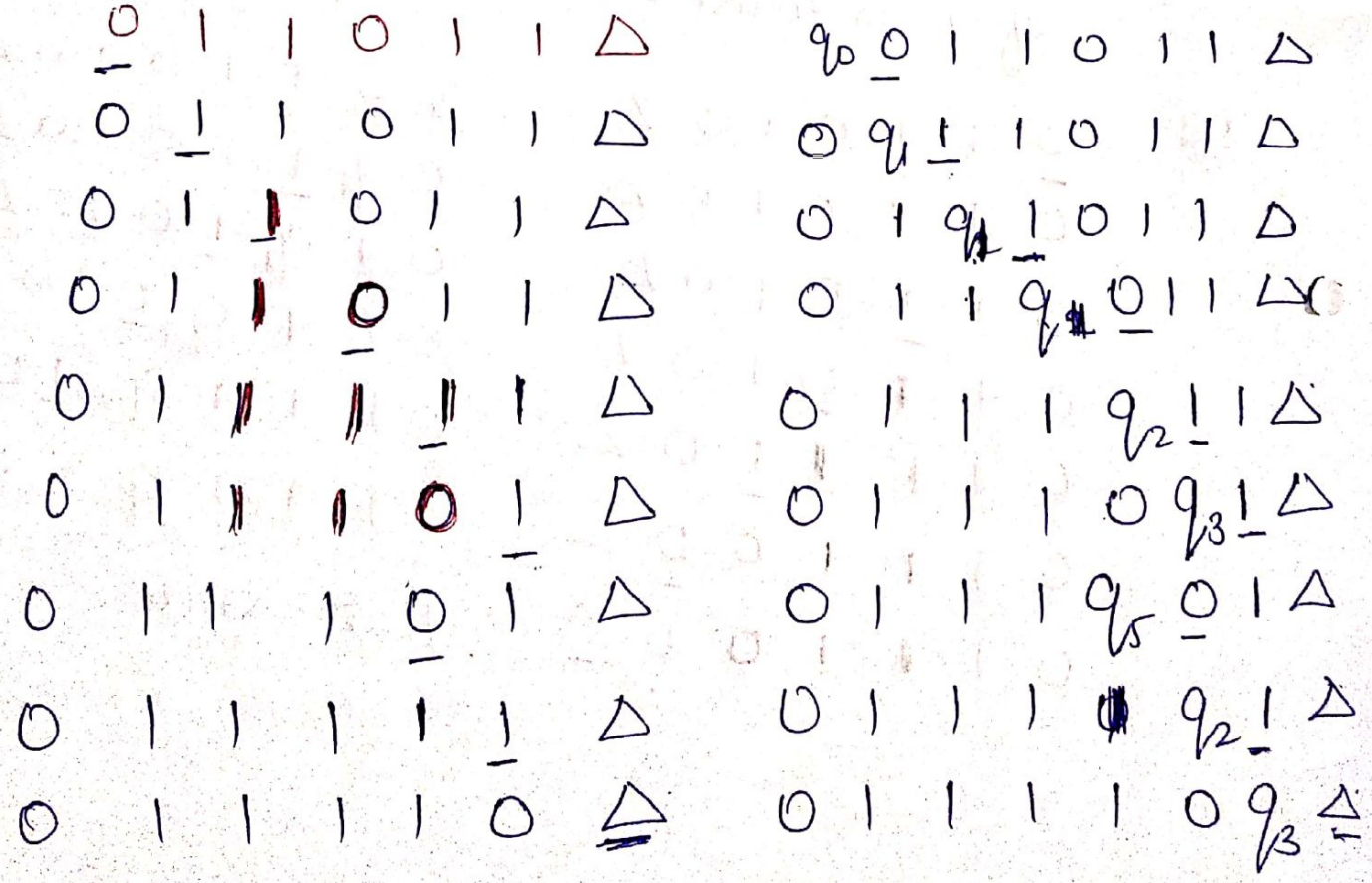
Case (iii) :- During searching pattern, if we found 1st 1 of pattern then write 1 & move in Right.

If found 0, after 1 i.e 2nd symbol of pattern then write 1 & move Right.

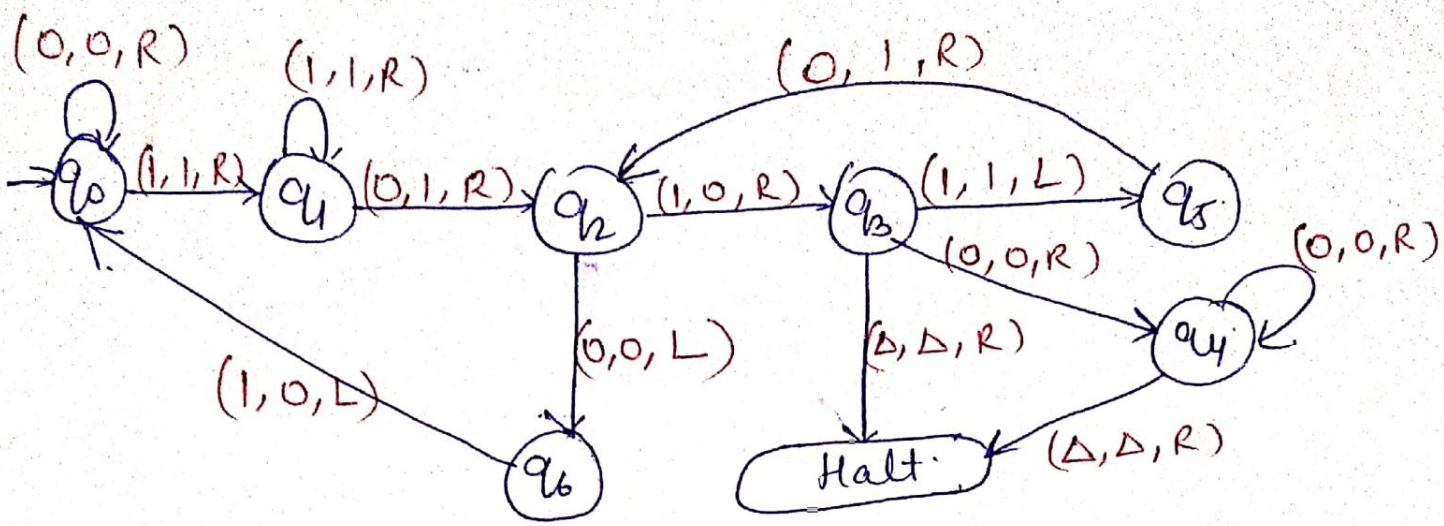
If found 1 i.e 3rd symbol of pattern after 0 then write 0 & move Right.

If after substring 101, 1 is found then move left. If left symbol is 0 then convert it to 1 & move Right. & again convert 1 to 0.

eg



Halt.



Suppose. If we have 0 1 1 0 0 0

0 1 1 0 0 0 Δ
0 1 1 0 0 0 Δ
0 1 1 0 0 0 Δ
0 1 1 0 0 0 Δ
0 1 1 1 0 0 Δ
0 1 1 1 0 0 Δ
0 1 1 0 0 0 Δ
0 1 1 0 0 0 Δ
0 1 1 0 0 0 Δ

q_0 0 1 1 0 0 0 Δ
0 q_1 1 1 0 0 0 Δ
0 1 q_1 1 0 0 0 Δ
0 1 1 q_1 0 0 0 Δ
0 1 1 1 q_2 0 0 Δ
0 1 1 q_6 1 0 0 Δ
0 1 1 0 q_0 0 0 Δ
0 1 1 0 0 q_0 0 Δ
0 1 1 0 0 0 q_0 Δ

In this sequence M/C will never Halt.