PAGE NO.:
DATE: / /

Theory of computation :—

Symbol :- a, b, 0, 1, ∓, ☆, ☨, ♀

↓ derives

"Σ" Alphbet — {a, b}, {a, b - - - z}
{0, 1}, {0, 1 - - - 9}

↓ makes

a String (collefion of Alphabet) :-

a, ab, aaa, abc, abb - - - -

Q) How many steings we can create using a alpha bet set?

Language :-
collection of valid words (steings)

for eg:-

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

a language L_1 = set of all string of length of "2".

$= \{aa, ab, ba, bb\}$ finite

L2 = Length "3" over Σ

= { aaa, aab, aba, abb, baa, bab, bba, bbb } finite

L3 = set of all string where every string starts with 'a'.

= {a, aa, ab, aab, aba, abb ----} infinite

Power of Σ :-

Σ = {a, b}

Σ^1 = set of all string over 'Σ' of length '1'

= {a, b}

Σ^2 = {aa, ab, ba, bb}

Σ^n = { all 'n' length strings over {a, b} }

Σ^0 = set of all string with length 0

= know as 'ε'

& $|\varepsilon| = 0$.

$\Sigma^* =$

$= \{$

$\Sigma^* =$

Take

Char

+

"C

Q

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \dots$

$= \{\varepsilon\} \cup \{a, b\} \cup \{ab, aa, ba, bb\} \cup \dots$

Σ^* = set of all string over $\{a, b\}$ (infinite)

Take the example of "C" lang :-

Char set :-

$\{a, b \dots z, A, B \dots Z, 0, 1 \dots 9, +, *, @ \dots\}$

```
void main () {
    int a, b;
    :
}
```

Program which is considered as string in ToC.

"C" prog. lang is set of all valid prog.

Q. How many prog. possible in "C" lag ?

Infinte

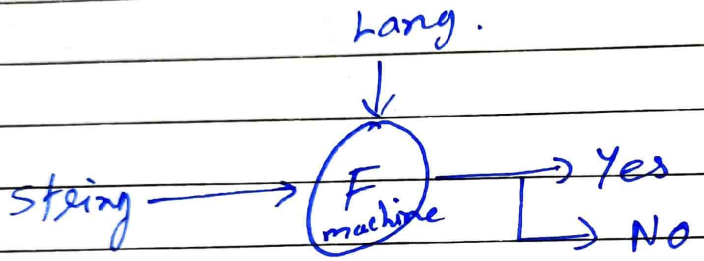
$= \{P_1, P_2, P_3 \dots\}$

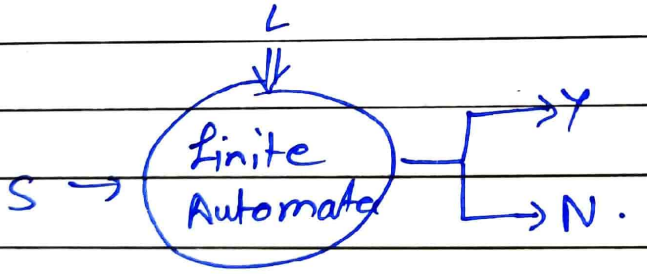
if we take any prog "P_n" it must belongs to the valid set.

The infinite no. of pray. we can't store in memory so "P_n" should be ralidted using some generic tool.

if Lang is finite the strings can be stored. But if it is infinite then it is not possible.

so machine should be like this.





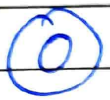
PAGE NO.:
DATE : __/__/__

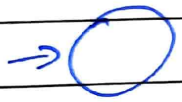
machine (The finite state machine) or finite Auto mata

 ⇒ state

→ ⇒ transition

a, b, 0, 1 ⇒ I/P symbols.

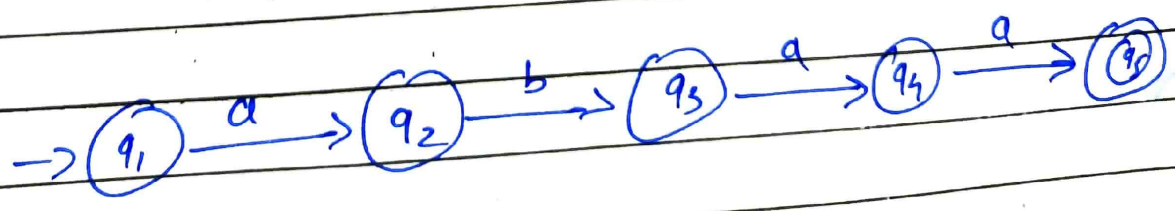
 ⇒ final state

 ⇒ initial state.

↶ → self loop transition.

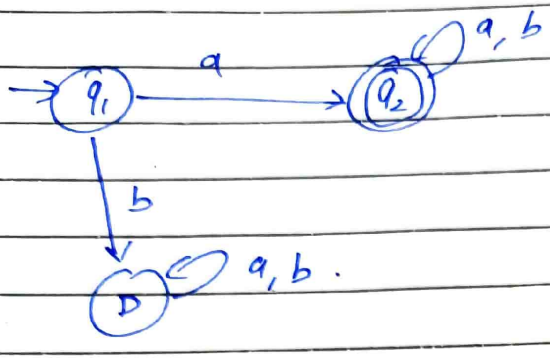
Q) set of all the strings which starts with 'a'

Q) Represent "abaa" over $\Sigma = \{a, b\}$



PAGE NO.:
DATE: / /

Q) L = set of all string which starts with 'a' over $\Sigma = \{a, b\}$.



Finite Automata :- (Types)

- FA with o/p
  - moore m/c
  - mealy m/c
- FA without o/p
  - DFA (deterministic Finite Auto.)
  - NFA (Non-deterministic —)
  - ε-NFA (ε — NFA)

DFA

Tuples

(

set of all

set of I/P sy

Transition

initial state

set of final

Trans

δ

DFA :-

Tuples :-

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

- Q → set of all states
- Σ → set of I/P symbols
- δ → Transition funct^n
- q_0 → initial state
- F → set of final state

Q = set of all states

Σ = set of I/p symbols

δ = Transition function

q_0 = initial state

F = set of all final state.

Transition function :-

δ :- $Q \times \Sigma \rightarrow Q$

$\{q_1, q_2, D\} \times \{a, b\} \rightarrow \{q_1, q_2, D\}$

$(q_1 \times a) \longrightarrow (q_2)$

$(q_2 \times b) \longrightarrow (q_2)$

$(q_1 \times b) \longrightarrow (D)$

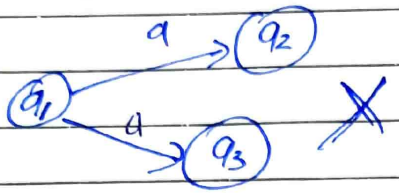
⋮

in DFA for a single I/p from a state it goes to only one next state.

in other words,

The DFA defines the next state with out any ambiguity for a single I/P.

Note: in DFA

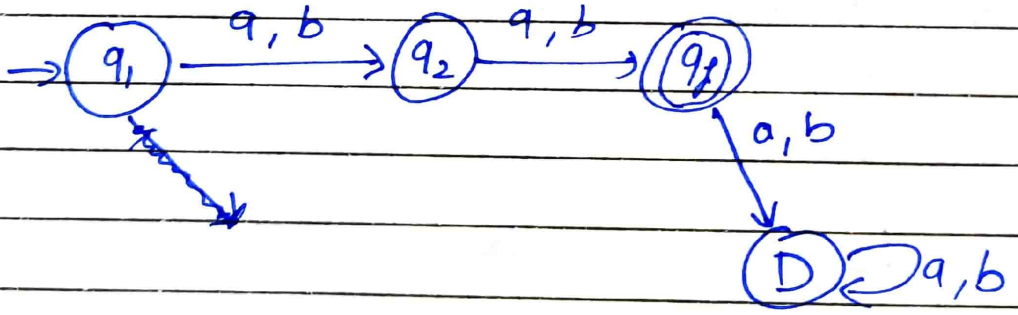


✗ this should not happen.

Q) Construct a DFA, that accept set of all string over $\{a,b\}$ of length 2.

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

$L = \{aa, ab, ba, bb\}$



The Final defⁿ of DFA

every state should constist out edges for m it with all the possible i/p & the with I/P symbol it ~~she~~ the next state shalld not be ambigous.

PAGE NO.:
DATE: __/__/__

String acceptance :-

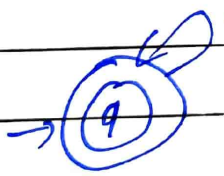
scan the entire string with the help of FA & we reach to the final state from initial state then string is accepted.

Lang. acceptance :-

A FA is said to accept a lang. if all the string in the lang are accepted & all the string not in the lang. should be rejected.

Rejection of non-valid strings are also important & it has the same weightage as acceptance.

for eg..

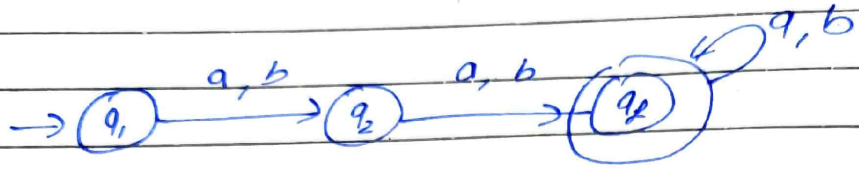


The given Automata accept every string which we throw on it. but it is not a valid Automata for the restricted strings. as it accept all the string - valid, invalid both.

DATE: / /

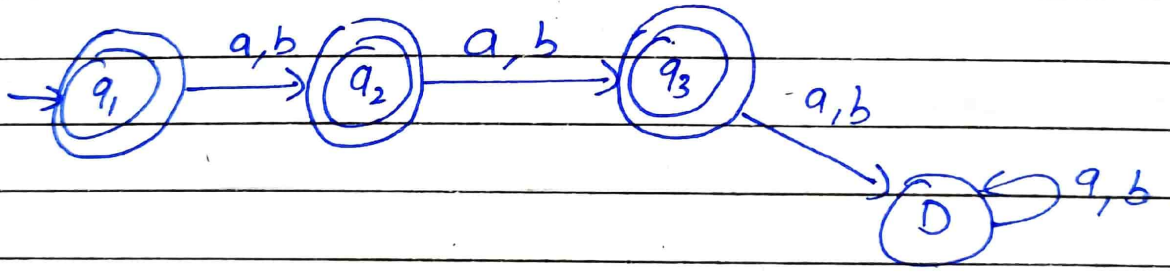
Q) Create a DFA over $\Sigma = \{a, b\}$ where $|w| \geq 2$ ~~(no. of words)~~ length of string must be "2" or greater.

$L = \{aa, ab, ba, bb, aaa, aba ----\}$



Q) DFA over ~~ab~~ $w = \{a, b\}$ where $|w| \leq 2$

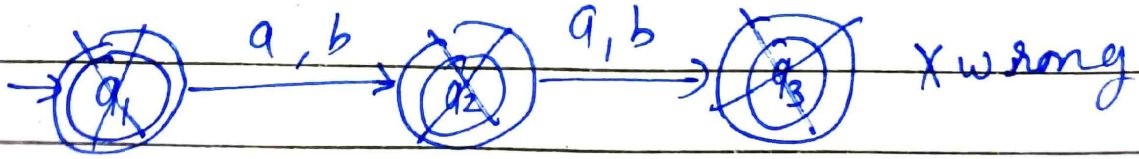
$L = \{\varepsilon, a, b, aa, ab, ba, bb\}$

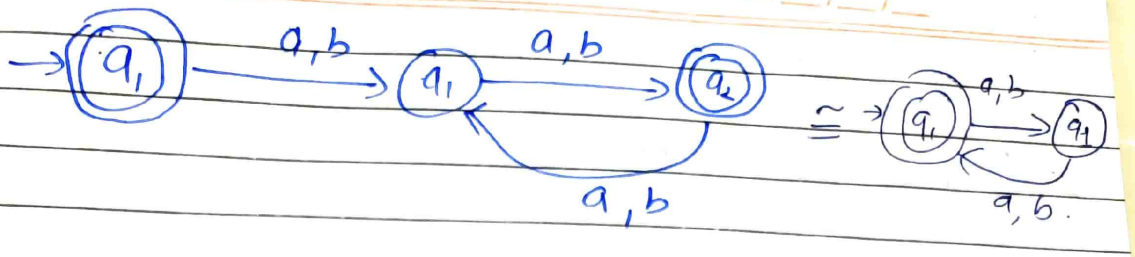


Q) DFA ~~Σ~~ $= \{a, b\}$ such that

$|w| \bmod 2 = 0$

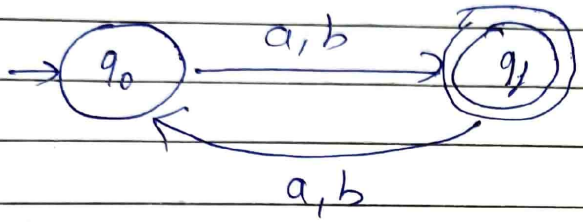
$L = \{\varepsilon, aa, bb, ab, ba, aaaa, bbbb ----\}$





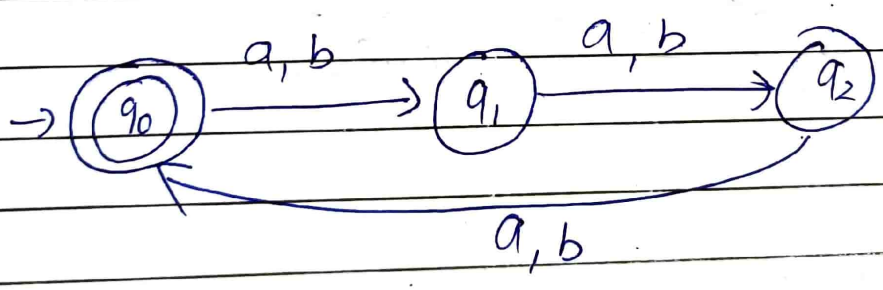
Q) DFA $w = \{a, b\}$ where

$|w| \bmod 2 = 1$.



Q) $w \in \{a, b\}^*$, $|w| \bmod 3 = 0$

$L = \{ \varepsilon, aaa, aab, aba, bbb ----- \}$



it is also written same as.

$|w| \cong \bmod 3$

OR

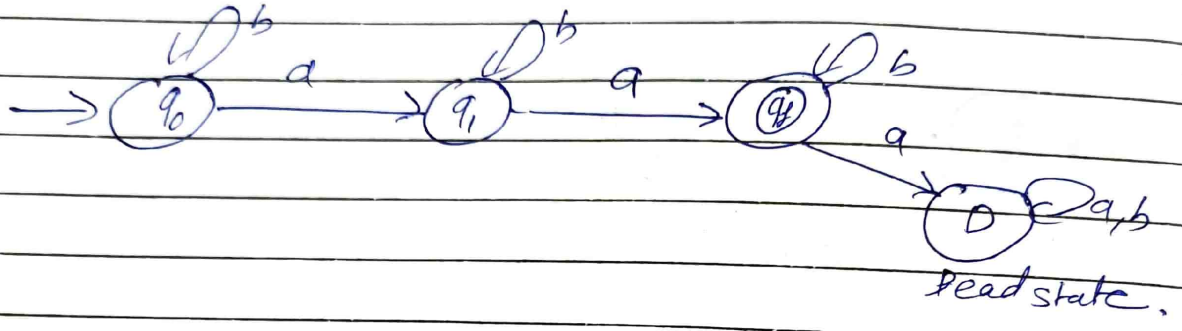
$|w| \bmod 3 = 0$

PAGE NO.:
DATE: / /

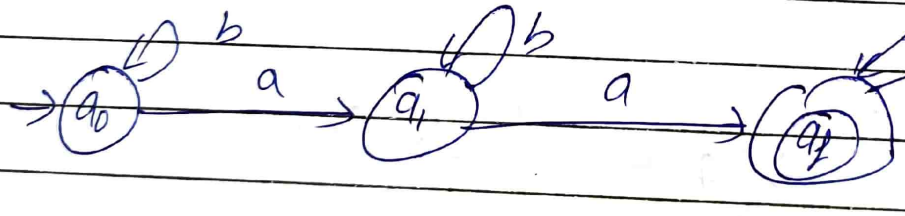
The $|w| \bmod n = 0$ DFA will have "n" no. of states

Q) $w \in \{a,b\}^*$, $n_a(w) = 2$

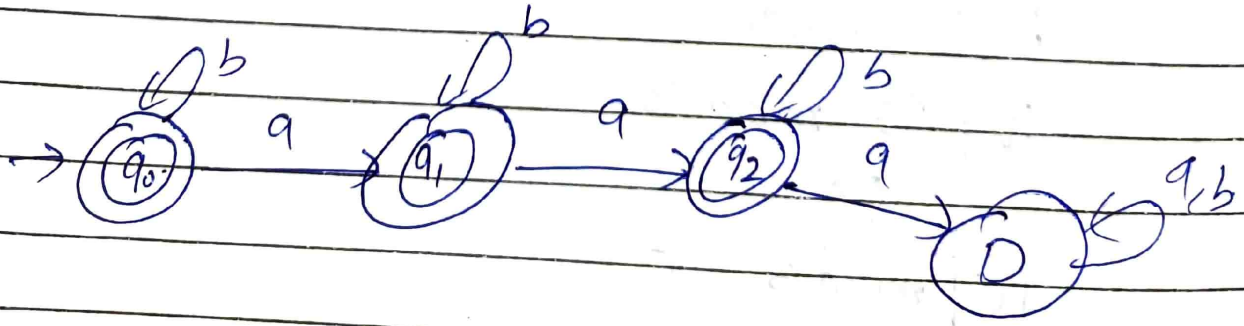
$L = \{aa, baa, aba, aab \ldots\}$



Q) $n_a(w) \geq 2$



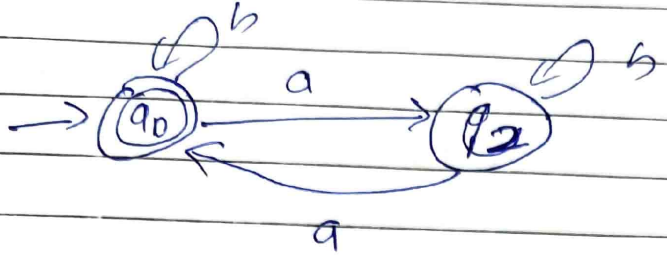
Q) $n_a(w) \not\geq 2$



Q) no. of 'a' must be even. over $\Sigma = \{a, b\}$

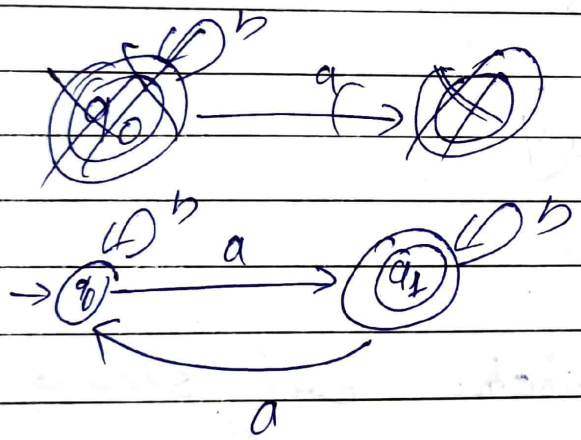
$w \in \{a, b\}^*$

$n_a(w) \bmod 2 = 0$ or $n_a(w) \equiv 0 \bmod 2$.



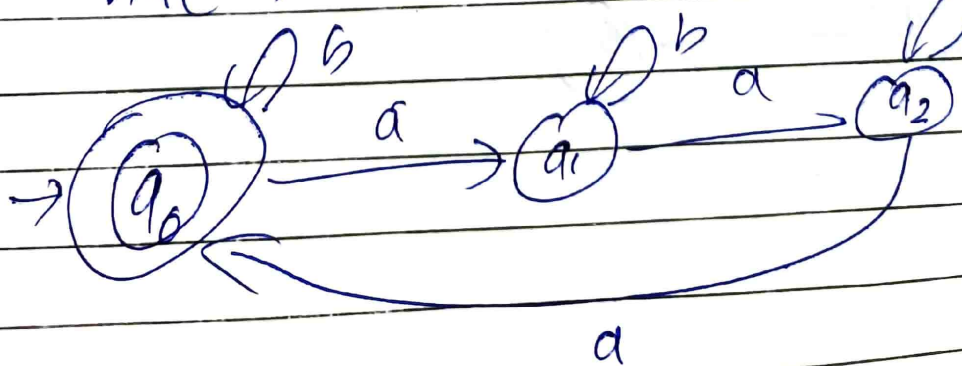
Q) $w \in \{a, b\}^*$

$n_a(w) \bmod 2 = 1$.



Q) $w \in \{a, b\}^*$

$n_a(w) \bmod 3 = 0$

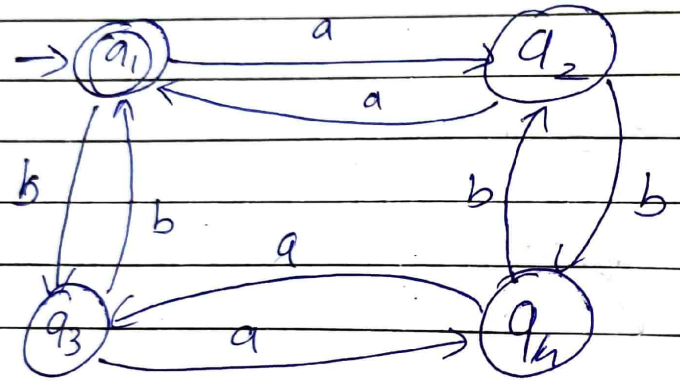


PAGE NO.:
DATE: / /

Q) minimal DFA $w \in \{a,b\}^*$

$$\begin{cases} n_a(w) \bmod 2 = 0 \\ n_b(w) \bmod 2 = 0 \end{cases}$$

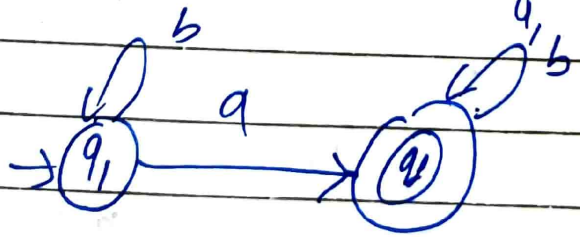
$L = \{\varepsilon, aa, bb, aabb, abab, baba, bbbb, aaaa \text{ - - - - - -}\}$



Q) $w \in (a,b)^*$ where each string contains 'a'

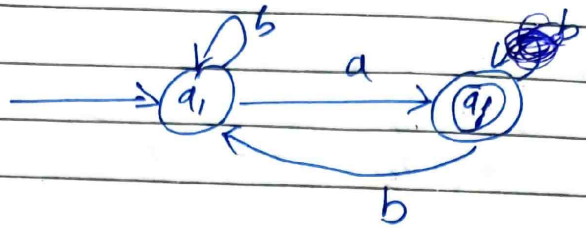
a

:-

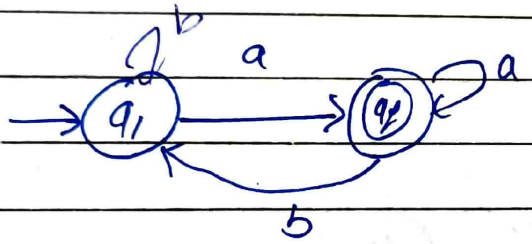


PAGE NO:
DATE: / /

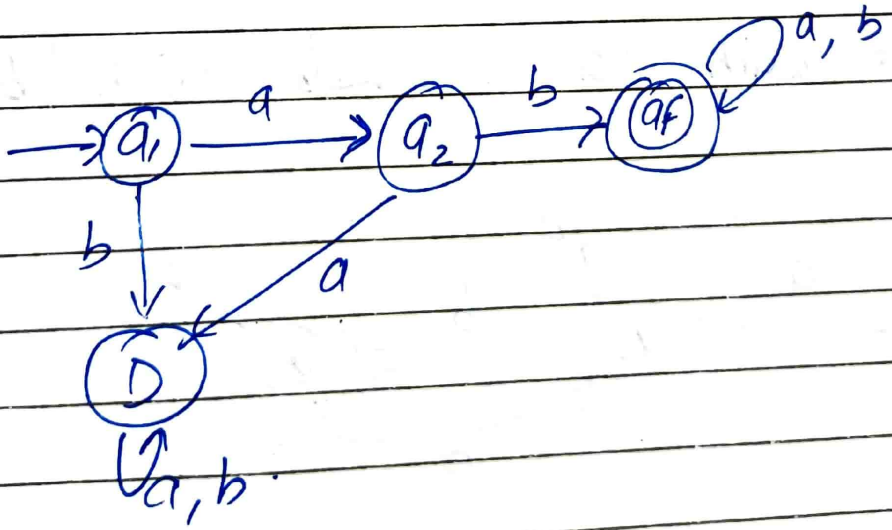
Q) set of all the strings over $w \in (a,b)^*$ that ends with 'a'



Q) starting with 'a' containing 'a' & end with 'a'.

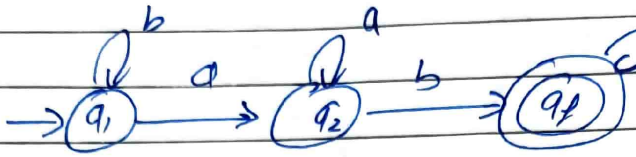


Q) starts with 'ab'

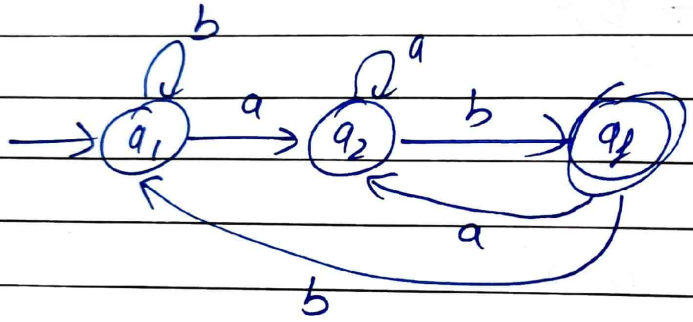


Q) Set of all string that containg 'ab' as substring.

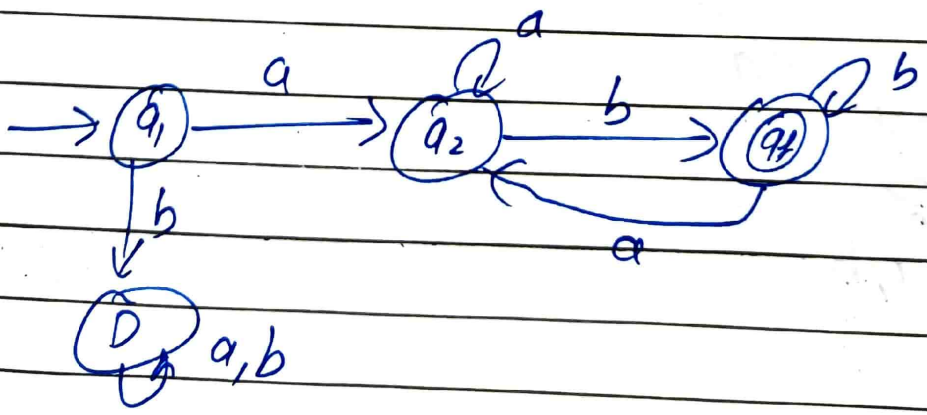
A.



Q) ends with 'ab'



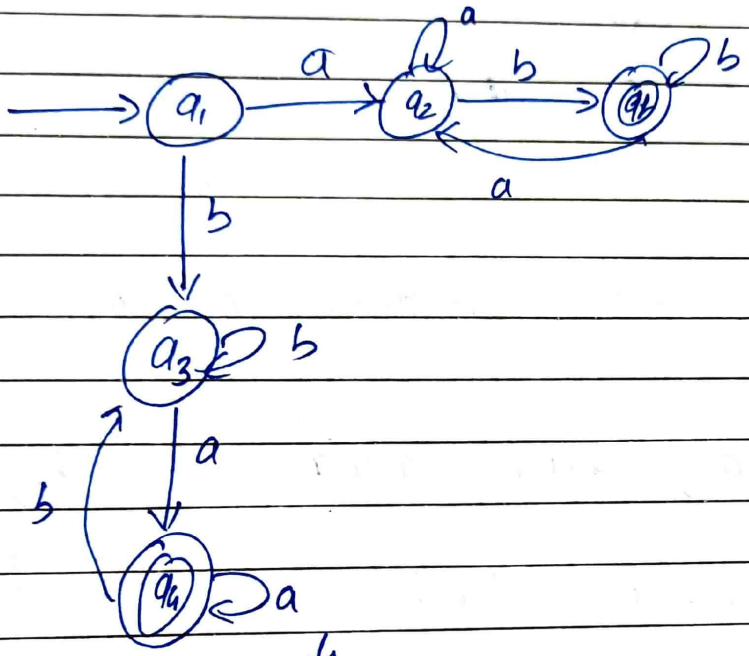
Q) starts with 'a' ends with 'b'



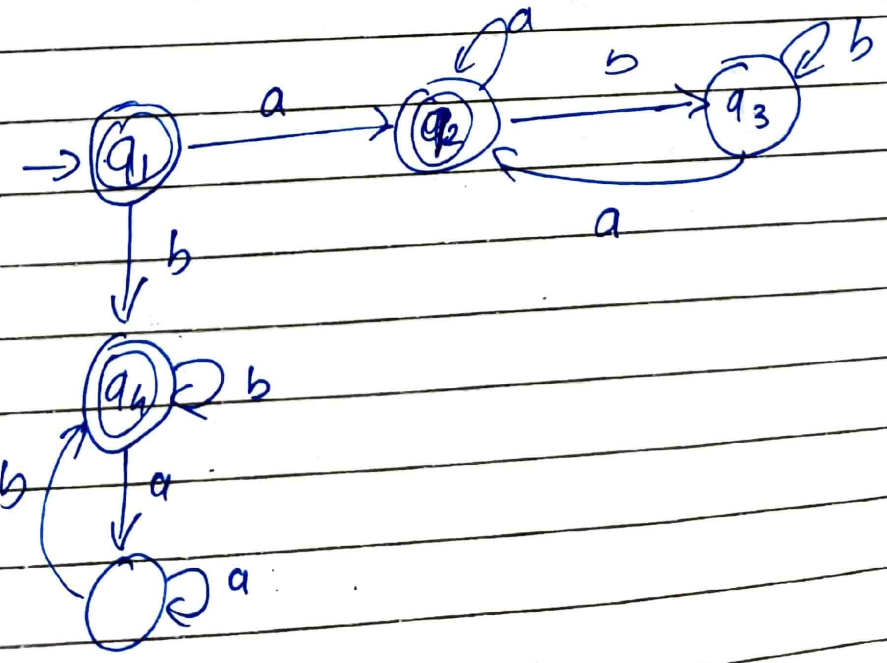
PAGE NO.:
DATE: / /

Q) Set of all string ~~which~~ ~~with~~ which starts & and ends with different symbols.

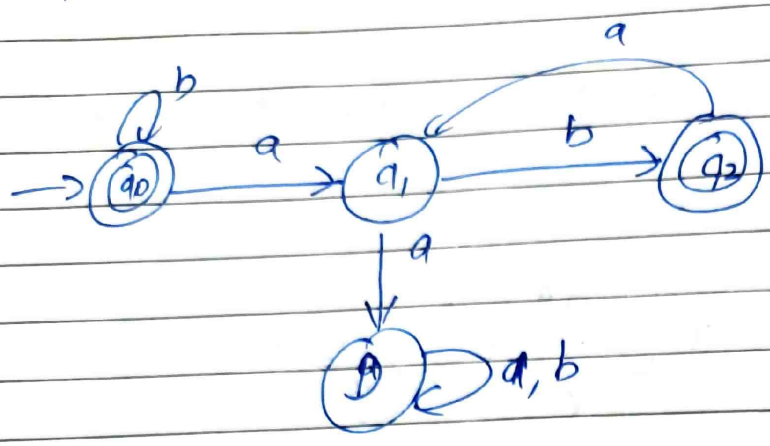
$L = \{ab, ba, abbb, aaab \ldots\}$



Q) starts and ends with same symbols. 18.



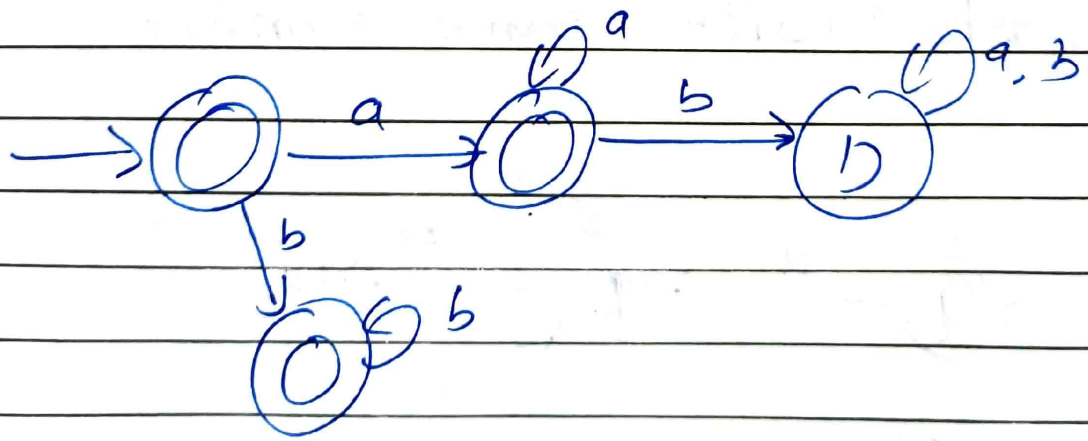
Q every 'a' should be followed by 'b' i.e ab



Q 'a' should not followed by 'b'

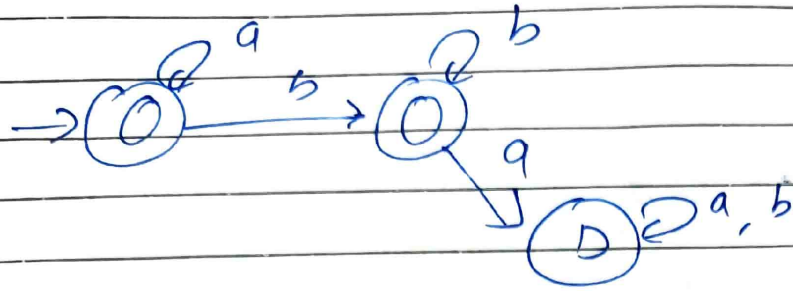
$L = \{\varepsilon, a, aaa, aaa \cdots, b, bb \cdots$

$ba, bba, bbba \cdots\}$



DATE: / /

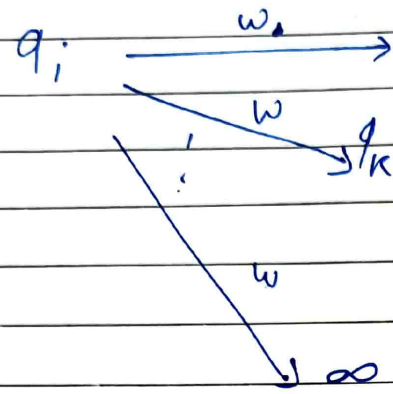
9) $L = \{a^n b^m \mid n, m \geq 0\}$



PAGE NO.:
DATE: __/__/__

NFA

$q_i \xrightarrow{w} q_j$ DFA.



Tuples :-

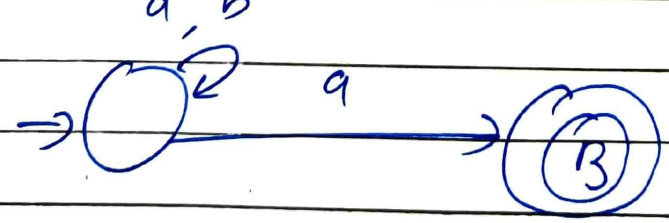
$(Q, \Sigma, \delta, q_0, F)$

all others are same as DFA.

$\delta :- \Sigma \times Q \rightarrow 2^Q$

2^Q = Power set

Q) ends with 'a'



PAGE NO.:
DATE: / /

What is 2^Q ?

$Q = \{A, B\}$, $\Sigma = \{a, b\}$.

$Q \times \Sigma$	2^Q
{A, a}	ϕ
{A, b}	{A}
{B, a}	{B}
{B, b}	{A, B}

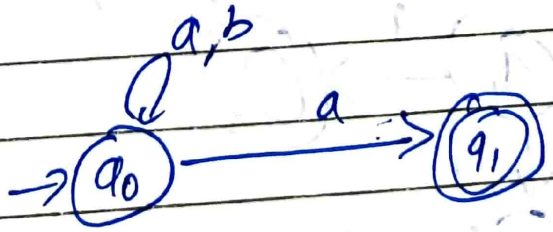
NFA vs DFA :-

NFA & DFA both are equally powerful That means what ever DFA can do the NFA can do too. & vice versa.

$$NFA \cong DFA$$

NFA to DFA Conversion :-

$L_1 = \{$ ends with an 'a' $\}$



State table :- (NFA)

	a	b
A	{A, B}	{A}
B	{ }	{ }

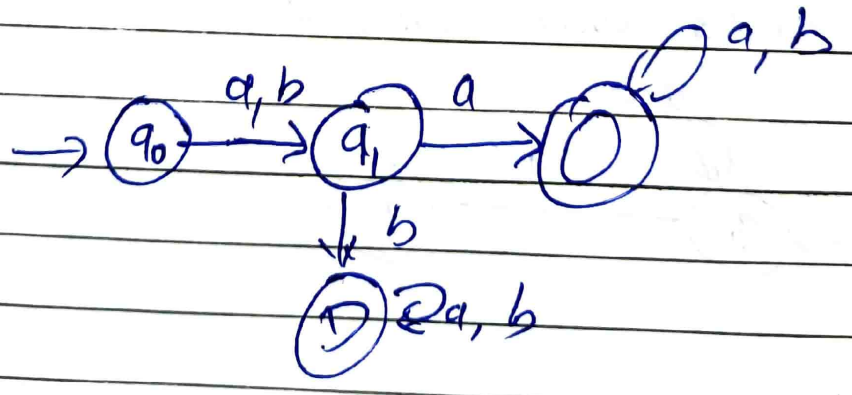
Conversion :-

	a	b
[A]	[AB]	[A]
[AB]	[AB]	[A]

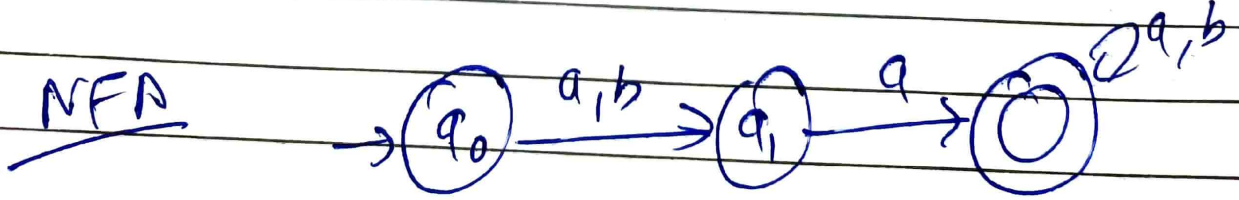
[AB]
↓
a new state
Single state

Q) "all string in which second symbol from RHS is 'a' ".

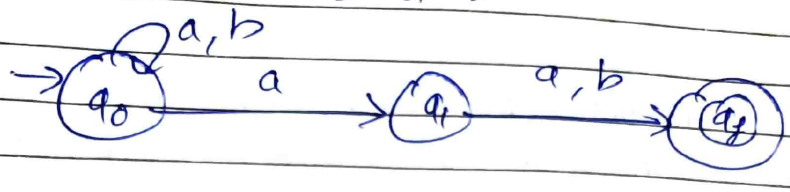
DFA



NFA



NFA Q) all the strings in which second symbol from RHS is "a".



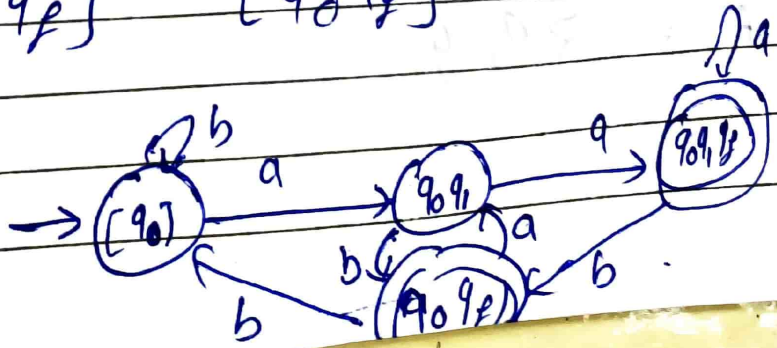
Ans :-

Table (NFA)

	a	b
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_f\}$	$\{q_f\}$
q_f^*	$\{\}$	$\{\}$

DFA table :-

	a	b
$[q_0]$	$[q_0 q_1]$	$[q_0]$
$[q_0 q_1]$	$[q_0, q_1, q_f]$	$[q_0 q_f]$
$[q_0 q_f]$	$[q_0 q_1]$	$[q_0]$
$[q_0 q_1 q_f]$	$[q_0 q_1 q_f]$	$[q_0 q_f]$



PAGE NO.:
DATE: / /

ε-NFA :-

A m/c without reading any I/p going to next state. It called "epsilon".

Tuples :-

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

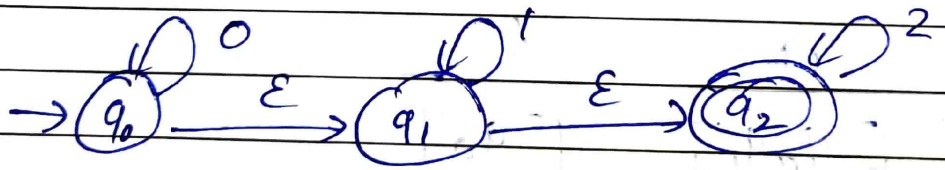
all the tuples are same except δ.

$\delta : \rightarrow$

$$Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^Q$$

Pow set

Example :-



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

$\Sigma = \{0, 1, 2\}$

$F = \{q_2\}$

Transiti

δ :-

States

q_0

q_1

q_2

$v = "\delta$

q_0

$\varepsilon -$

PAGE NO.:
DATE: / /

Transition table :-

δ :-

States	0	1	2	ε
q_0	$\{q_0\}$	ϕ	ϕ	$\{q_1\}$
q_1	ϕ	$\{q_1\}$	ϕ	$\{q_2\}^*$
q_2	ϕ	ϕ	$\{q_2\}$	ϕ

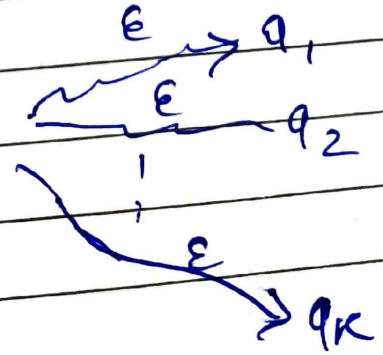
$w =$ "002"

$q_0 \xrightarrow{0} q_0 \xrightarrow{0} q_0 \xrightarrow{\varepsilon} q_1 \xrightarrow{\varepsilon} q_2 \xrightarrow{2} (q_2)^*$

So "w" is acceptable by ε-NFA.

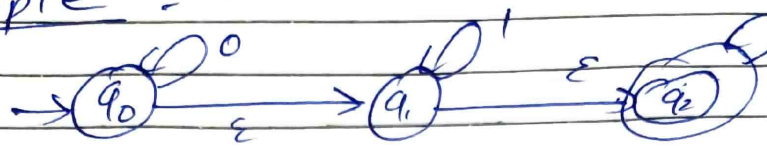
ε-closure of a state :-

ε-closure(q_i) = { all the states $q_1 \cdots q_x \cdots$ that can reachable from q_i using ε }



PAGE NO.:
DATE: / /

Example :-



ε - closure $(q_0) = \{q_0, q_1, q_2\}$

remember $\delta(q_0, \varepsilon) \rightarrow q_0$

ε - closure $(q_1) = \{q_1, q_2\}$

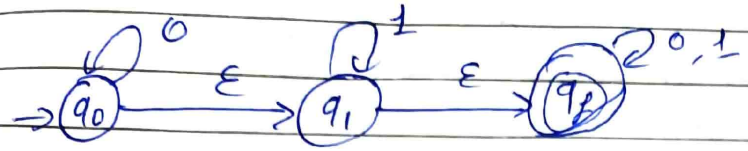
ε - closure $(q_2) = \{q_2\}$

Extended Transition function :-

$Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^Q$

PAGE NO:
DATE: __/__/__

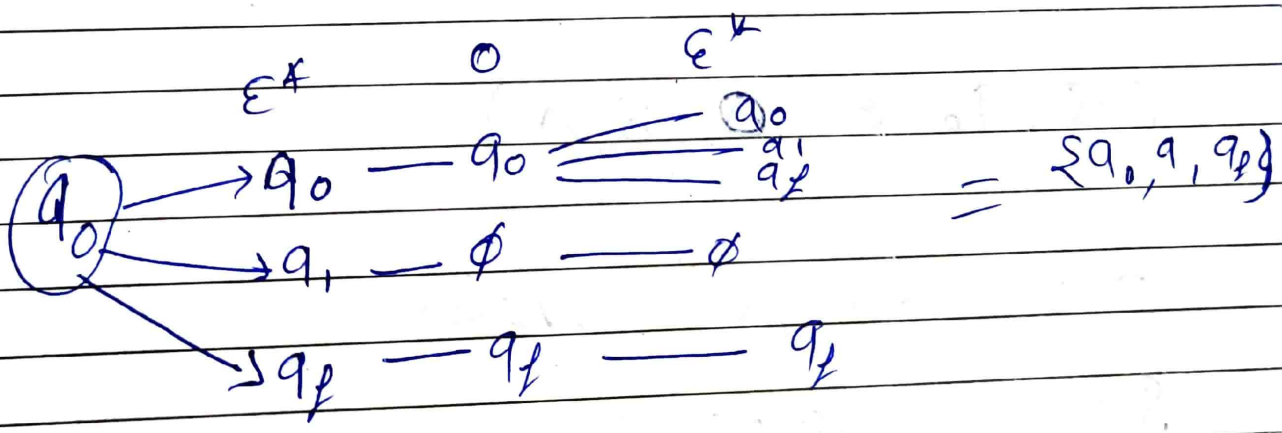
ε-NFA to NFA :-

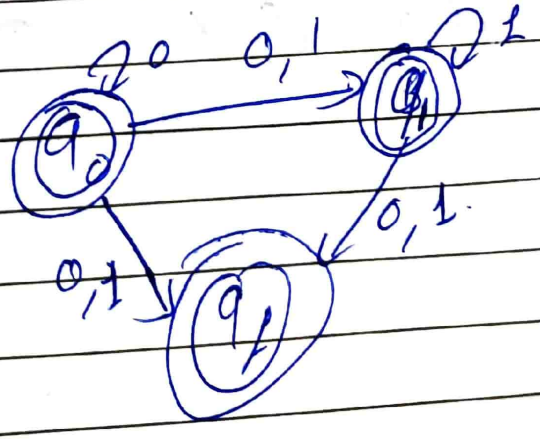


Sol^n.

Transition table

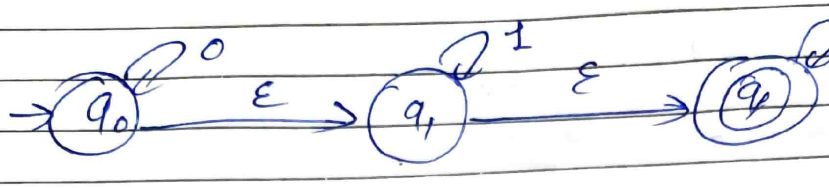
	0 ($\varepsilon^* 0 \varepsilon^*$)	1 ($\varepsilon^* 1 \varepsilon^*$)
q_0	$\{q_0, q_1, q_f\}$	$\{q_1, q_f\}$
q_1	$\{q_f\}$	$\{q_1, q_f\}$
q_f	$\{q_f\}$	$\{q_f\}$





PAGE NO.:
DATE: / /

Q.



Mealy & Moore m/c s :-

In the mealy m/c the o/p depends on current state & current I/Ps.

6 ~~tupp~~ tuples :-

$(Q, q_0, \Sigma, O, \delta, \lambda)$

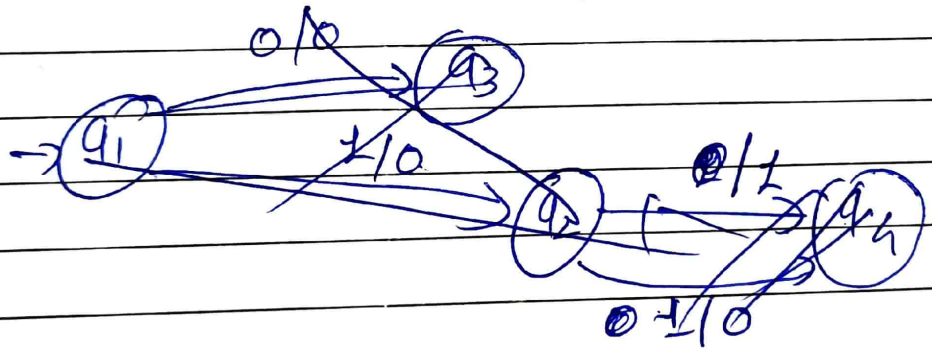
O :- out put alphabet

δ :- $Q \times \Sigma \to Q$.

λ = output function $Q \times \Sigma \to O$

Table for mealy m/c :-

Present state	next state			
	a=0		a=1	
	state	o/p	state	o/p
q_1	q_3	0	q_2	0
q_2	q_0	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0



moore m/c :-

Tuple :-

$(Q, \Sigma, O, \delta, \lambda, \delta)$

$O \rightarrow$ output alphabet

$\lambda \rightarrow Q \times \Sigma \rightarrow O$

DATE: / /

moore

present state	next states		output
	a=0	a=1	0
q_0	q_3	q_1	0
q_1	q_1	q_2	1
q_2	q_2	q_3	0
q_3	q_3	q_0	0

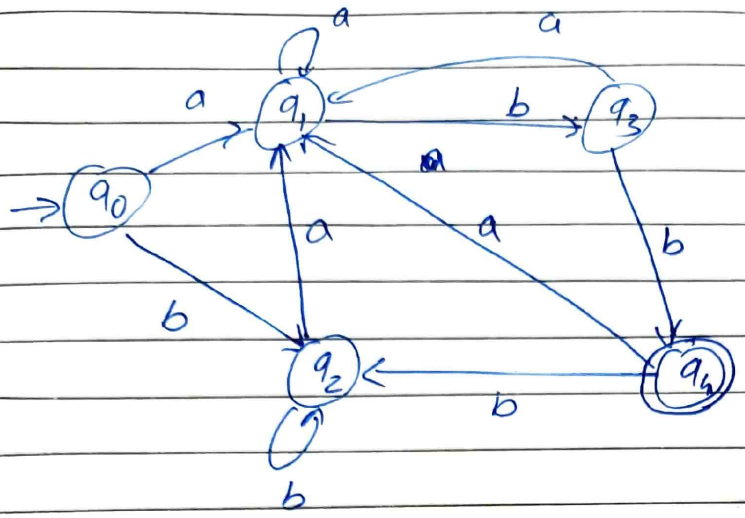
Mini

→ (q0

Tu
if
St

PAGE NO :
DATE : __/__/__

Minimization :-



~~Ex~~

Two states (P, q) called to be equivalent if both are ~~final~~ final states or non final states.

$$\left.\begin{array}{l} \delta(P, w) \in F \\ \& \\ \delta(q, w) \in F \end{array}\right\} \text{ equivalent.}$$

$$\left.\begin{array}{l} \delta(P, w) \notin F \\ \delta(q, w) \notin F \end{array}\right\} \text{ equivalent.}$$

PAGE NO.:
DATE : / /

Soln :-

	a	b
$\rightarrow q_0$	q_1	q_2
q_1	q_1	q_3
q_2	q_1	q_2
q_3	q_1	q_4^*
q_4^*	q_1	q_2

~~0 - equivalence states (set)~~

0 - equivalence set of states (Π_0)

φ_1^0 = { final states' set }

φ_2^0 = { non final states' set }

$\varphi_1^0 = \{q_0, q_1, q_2, q_3\}$

$\varphi_2^0 = \{q_4\}$

$\Pi_0 = \{\varphi_1^0, \varphi_2^0\}$

Π_1 (set of ϕ^1)

$\phi_1^1 = \{q_0, q_1, q_2\}$

$\phi_2^1 = \{q_3\}$

$\phi_3^1 = \{q_4\}$

$\Pi_1 = \{\phi_1^1, \phi_2^1, \phi_3^1\}$

Π_2 (set of ϕ^2)

$\phi_1^2 = \{q_0, q_2\}$

$\phi_2^2 = \{q_1\}$

$\phi_3^2 = \{q_3\}$

$\phi_4^2 = \{q_4\}$

PAGE NO.:
DATE: / /

$\pi_3 =$

$\varphi_1^3 = \{q_0, q_2\}$

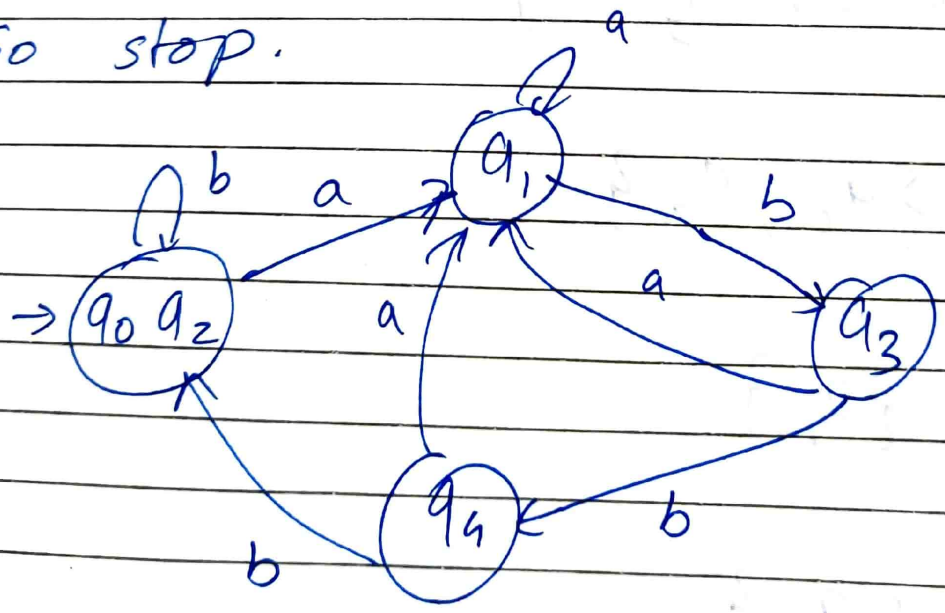
$\varphi_2^3 = \{q_1\}$

$\varphi_3^3 = \{q_3\}$

$\varphi_4^3 = \{q_4\}$ *

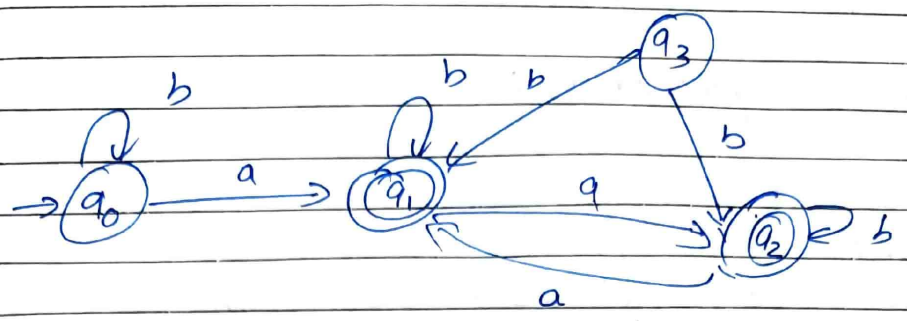
$\pi_2 = \pi_3$

So stop.

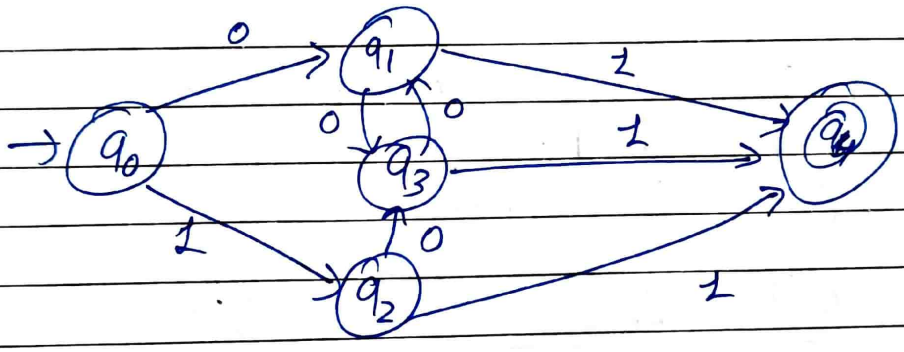


PAGE NO.:
DATE: / /

Q



Q)



Q