Automata Theory

PREPAIRED BY – DEEPAK
A.P in CSE, PIET

Introduction:

- Automata theory is the study of abstract computing devices or "machines".
- Before there were computer, in 1930, (Alen Turing) introduced an abstract machine that has all the capabilities of todays computer.

History of Automata Theory:-

- In 1930, Alan turing studies Turing Machines. which Calculate the "Halting Problem".
- In between 1940 – 1950, the finite Automata machine studied & then Noam Chomsky proposes the "Chomsky Hierarchy" for formal language.
- In 1969, cook introduce "Intractable" Problems or "NP-Hard" Problems.
- In 1970, modern Computer Science tools such as Compiler,

Languages & Grammars

Languages:- "A language is a collection of sentences of finite length all constructed from a finite alphabet of symbols".

→ An alphabet is a set of symbols: $\{0, 1\}$

→ A sentence or word are strings of symbols:

If $\Sigma = \{0, 1, 00, 01, 10, 1 - \ldots\}$

A language is a set of sentences:

If $L = \{000, 0100, 0010, \ldots\}$

→ A grammar is a finite list of rules defining a language.

$S \to 0A$ $\quad B \to 1B$

$A \to 1A$ $\quad B \to 0F$

$A \to 0B$ $\quad F \to \epsilon$

→ **Grammer:-** "A grammar can be regarded as a device that enumerates the sentences of a language"

1) Symbol → a, b, 0, 1 - - - -

2) Alphabet → "Σ" → {a,b} or {a, b, c}, {0,1} - - -

↘ "finite set of symbols"

3) String → Some length of Alphabet is called string.

f.e {a, b, aa, ba, bc - - - }

String of length '2' over alphabet {a, b} is

{aa, ab, ba, bb}

4) Language : Collection of Strings

f.e: Σ = {a, b}

L_1 = Set of all strings of length '2'.

= {aa, ab, ba, bb}.

L_1 is a finite language because the set is finite.

f.e: L_2 = Set of all strings of length 3.

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

f.e. L_3 = Set of all strings where each string start with 'a'.

$= \{a, aa, aaa, ab, aba, abbb \text{ ---} \}$

L_3 is infinite language.

Powers of Σ :-

If $\Sigma = \{a, b\}$ $\therefore$ Σ^0 = set of string of length zero. $\{\in\}$.

Σ^1 = set of all strings over Σ of length 1.

$= \{a, b\}$.

$\Sigma^2 = \Sigma . \Sigma = \{a,b\}\{a,b\} = \{aa, ab, ba, bb\}$.

$\Sigma^3 = \Sigma . \Sigma . \Sigma = \{a,b\}\{a,b\}\{a,b\} = \{$ 8 items. $\}$

$|\Sigma^3| = 8$

Σ^n = n length strings

$\phi = \{\}$ nullset
$\in$ = nullstring

$\rightarrow \Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \text{ ---}$

$\{\in\} \cup \{a,b\} \cup \{aa, ab, ba, aa\}$

$= \{\in, a, b, aa, ab, ba, aa, aab, aba, \text{ ---} \}$

means Σ^* contain all the possible combinations over $\Sigma = \{a,b\}$ $\therefore$ it is called Universal set & infinite in nature.

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

$L_1 = \{aa, ab, ba, bb\}$

$S = aaa$ (check)

If L_1 is finite we can easily check the string is in language or not

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

$L_2 = \{a, aa, aaa, aba \cdots\}$

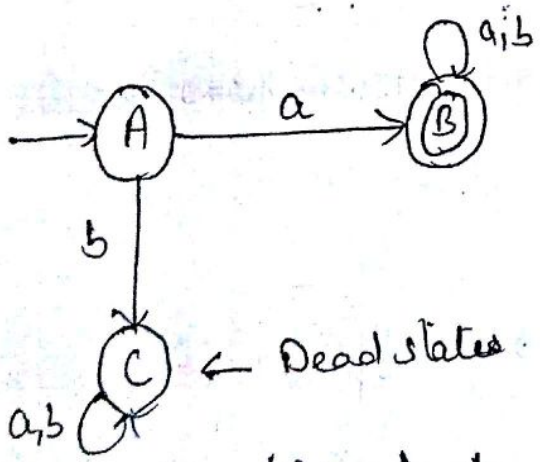
$S = abbaa$

If L_2 is infinite then it is very difficult to a say string is accpt or not.

So to overcome the infinite language problems we design the finite Automata machine.

Ex L_1 = set of all strings which start with 'a'

$\{a, aa, ab, aaa \cdots\}$



← Dead state (Finite Automata M/C)

Now, with this finite machine we can check any string whether it is accepted or not.

f. E the string s= aab , Now we check it in the finite machine.

After checking the machine accept the string a 'aab'.

check the string s= babab

This string is rejected by the machine as it is start with 'b' symbol. which takes the machine in the dead state.

Note!-

String Acceptance :- Scan the entire string, if we reach a final state from the initial state.

Language Acceptance :- A language is said to be accepted if finite Automata accept all the strings in the language. & reject all the string which are not in the language.

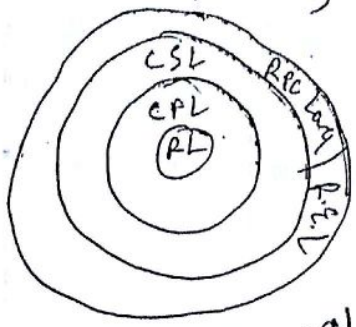
→ A Chomsky Hierarchy is a classes of formal languages:

① Regular (DFA)

② Context-free (PDA) — DPDA; NDPDA (more powerfull than DPDA)

③ Context-Sensitive (LBA) (Linear Bounded Automata)

④ Recursively-enumerable (TM)



Strings:-

A string or word is a finite sequence of symbols chosen from Σ.

Empty string is ϵ.

Length of string w, denoted by $|w|$, is equal to the number of characters in string.

E.g $x = 0|0|00$ $\quad |x| = 6$

Σ^1 = set of all string of length 1. $\{a, b\}$

Σ^2 = String of length 2 as.

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

$\Sigma^3 = \Sigma\Sigma\Sigma = \{a,b\}\{a,b\}\{a,b\} = 2^3 = 8$

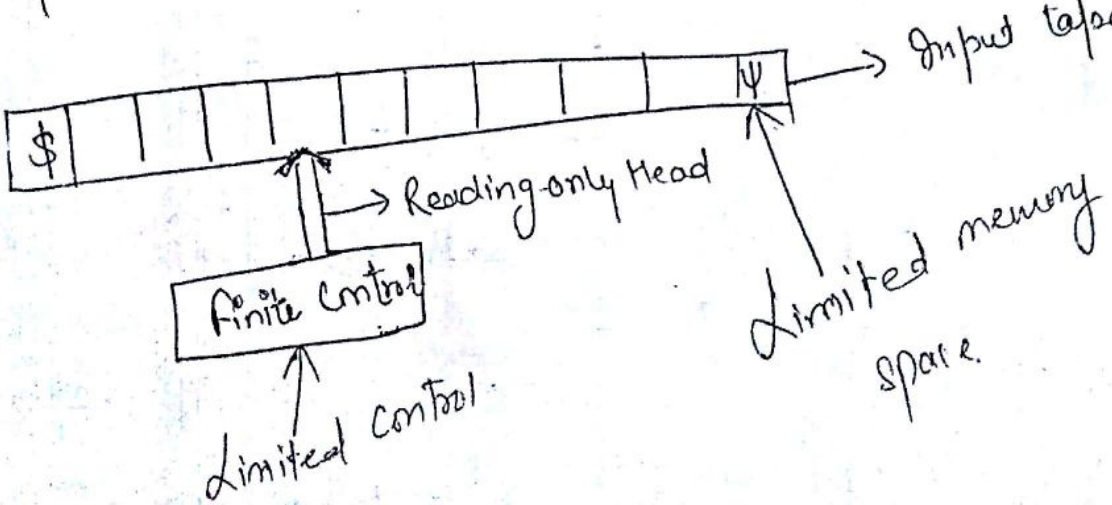
Finite Automata :-

→ A finite Automata is called "finite" because number of possible state and number of letter in the alphabet are both finite, and "automation" because the change of the state is totally given by the input.

→ Finite Automata is Abstract model of computer system.

→ The meaning of Abstract is that theoretically it is closest to the description of a computer system & formally describe the behaviour of computer system.

→ The finite Auto mata is shown as :→



→ The finite Auto mata has input tape, read-only Head, and finite control

In figure, the input is provided on the tape & head reads one symbol on the ~~finite control~~ tape & move ~~forwarded~~.

The movement is decided by the finite control.

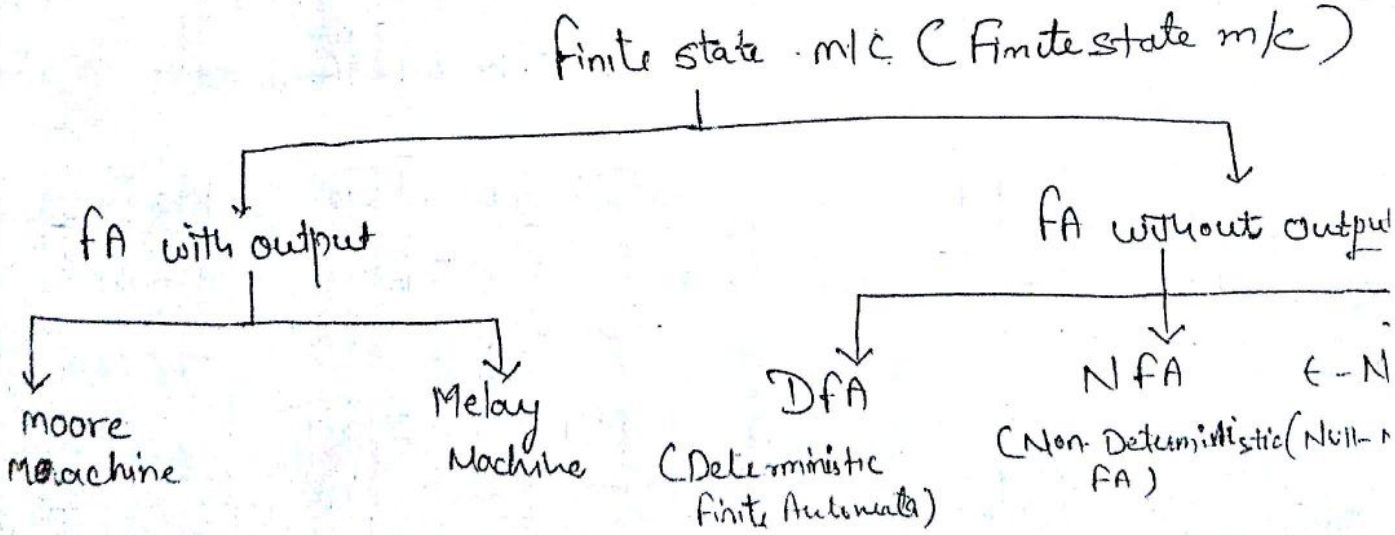
When the input is exhausted, the finite automata decides the validity of the input by acceptance or rejection.

Disadvantage :-

① It does not write anything.

② Finite Automata (FA) is also called finite state automata (FSA) or finite state machine (FSM).

TYPES of FSM (Finite State M/C)



① DFA (Deterministic Finite Automata) :->

→ A DFA 'M' is described by 5-tuples as:

$(Q, \Sigma, \delta, q_0, f)$ where.

1) Q = finite, non-empty set of state

2) Σ = an input alphabet (non-empty and finite)

3) δ = is transition function which maps $Q \times \Sigma \to Q$ i.e the head read a symbol in its present state and moves into next state.

4) $q_0 \in Q$ = known as intial state

5) $f \subseteq Q$ = known as set of final states

TRANSITION GRAPH :-

→ A transition graph is a collection of three things:-

① A finite set of states, at least one of which is known as initial state represent as →○.
& some of which are finial states represented as →◎

2) (2) An alphabet Σ of possible input letters from which input strings are formed.

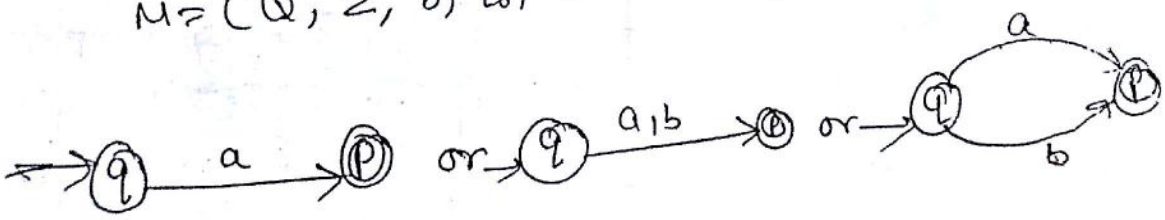
(3) A finite set of transitions (edges) that show how to go from some states to some others, based on reading specified substrings of input letters.

NOTATIONS for DFA :-

→ There are 2 notations for describing automata:

1) Transition Diagram
2) Transition table

1) Transition Diagram :- A transition diagram for DFA $M = (Q, \Sigma, \delta, q_0, f)$ is a graph defined as:



2) Transition table :- It is a tabular representation of a function like δ that takes two argument & returns a state.

→ The rows of table correspond to the states and the columns correspond to the input.

f. E :-

row = states
column = input

δ / Σ	a	b
→q_0	q_1	q_2
q_1	q_2	q_0
*q_2	q_2	q_2

q_0 is intial state & q_2 is final state.

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

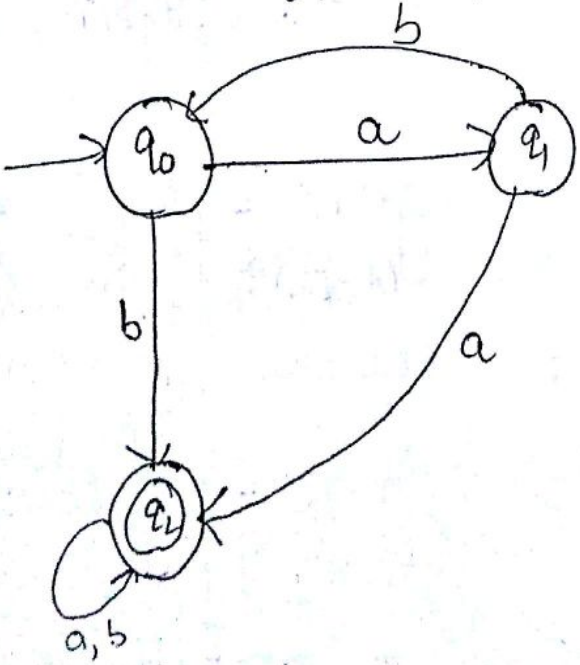
$f = \{q_2\}$

$\delta(q_0, a) = q_1$, $\delta(q_0, b) = q_2$

$\delta(q_1, a) = q_2$, $\delta(q_1, b) = q_0$

$\delta(q_2, a) = q_2$, $\delta(q_2, b) = q_2$

The transition diagram for above transition table is.



NFA (Non-Deterministic finite Automata)

A NFA of machine M is described by 5-tuple as:

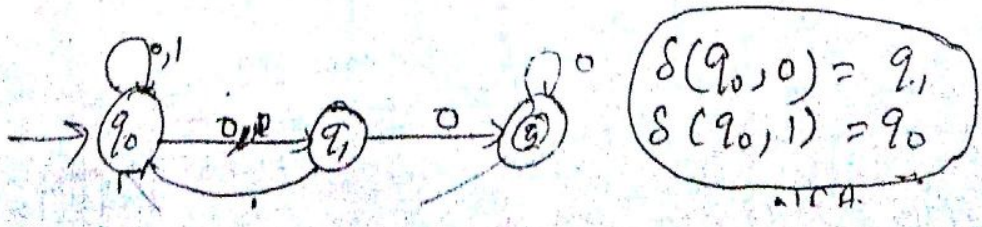
$(Q, \Sigma, \delta, q_0, F)$, where

1) Q is finite, non-empty set of states. It contain all the states.
2) Σ is an input alphabet.
3) δ is transition function which maps $Q \times \Sigma = 2^Q$

   2^Q is powerset of Q.
4) $q_0 \in Q$ known as intial state
5) $F \subseteq Q$ known as final state..

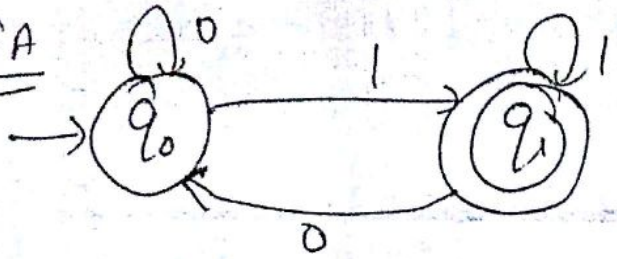
The Difference between NFA & DFA is only in transition functions. In DFA transition function maps on at most one state & in NFA transition fun" maps on at least one state for an input symbol.

NFA Example:- accept a string end with 00.



DFA & NFA Examples:

DFA



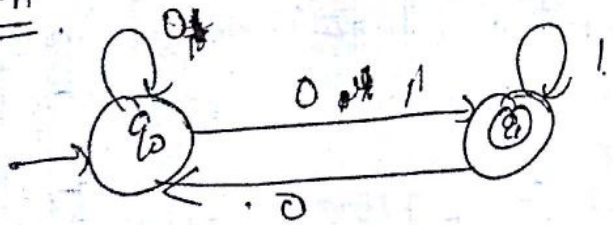
$\delta(q_0, 0) = q_0$
$\delta(q_0, 1) = q_1$
$\delta(q_1, 0) = q_0$
$\delta(q_1, 1) = q_1$

$Q = \{q_0, q_1\}$

& $Q \times \Sigma = Q$

All the outcomes states come from the Q states as $\{q_0, q_1\}$.

NFA



$\delta(q_0, 0) = \{q_0, q_1\}$
$\delta(q_0, 1) = [q_1]$
$\delta(q_1, 0) = [q_0]$
$\delta(q_1, 1) = [q_1]$

$Q = \{q_0, q_1\}$

$2^Q = \{\epsilon, \{q_0\}, \{q_1\}, \{q_0, q_1\}\}$

$2^2 =$ 4 states

Algorithm to Convert NFA to DFA →

Problem statment:-

Let $X = (Q_X, \Sigma, \delta_X, q_0, f_X)$ be an NDFA which accepts the language $L(X)$. we have to design an equivalent DFA $Y = (Q_Y, \Sigma, \delta_Y, q_0, f_Y)$ such that $L(Y) = L(X)$.

Algorithm:-

Input :- An NDFA

output :- Equivalent DFA

Step 1 Create state table from the given NDFA

Step 2 Create a blank state table under possible input alphabets for the equivalent DFA.

Step 3 Mark the start state of the DFA by q_0

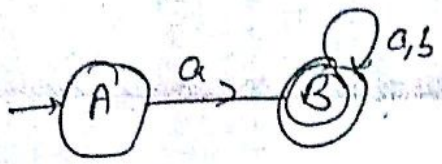
Step 4 Find out the combination of states $\{Q_0, Q_1 \cdots Q_n\}$ for each possible input alphabet.

Step 5: Each time we generate a new DFA state under the input alphabet columns, we have to apply step 4 again, otherwise go to step 6.

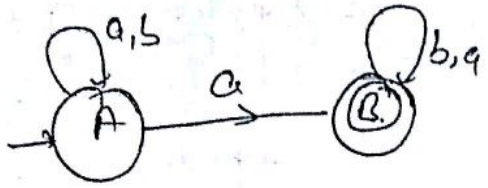
Step 6: The states which contain any of the final states of the NDFA are the final states

NFA Examples:-

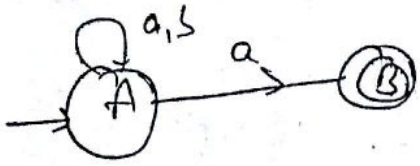
1) accept a string start with 'a'.



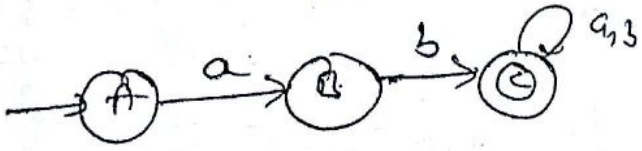
2) Accept a string Contains 'a'



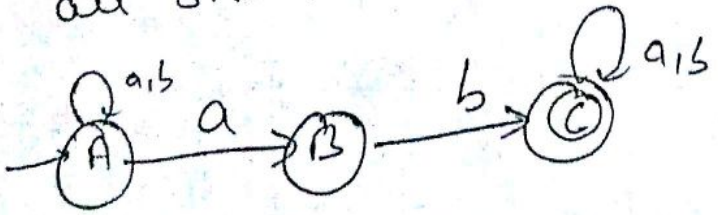
3) Accept a string End with 'a'



4) all string start with 'ab'



5) all string contains ab as substring



Comparision between DFA and NFA

DFA

1) DFA stands for deterministic finite Automata

2) for an input, there is only single Transition

3) There is no null move

4) In DFA, $Q \times \Sigma = Q$

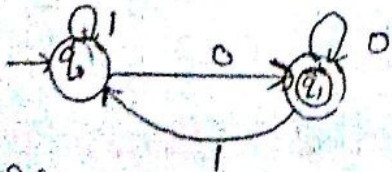
5) Execution time to execute a input string is ~~more~~ less as compare to NFA

6) For automation purpose DFA's are used

7) max of n states

8) Power is same.

9) Example: DFA ending with 0.



$\delta(q_0,0) = q_1 \quad \delta(q_0,1) = q_0$

NFA.

1) NFA stands for non-deterministic finite Automata

2) for an input, there can be multiple transitions.

3) There can be null moves.

4) In NFA, $Q \times \Sigma = 2^Q$ where 2^Q is power set of Q.

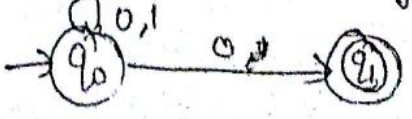
5) Execution time to execute a input string is more as compare to DFA.

6) NFA are not used for automation.

7) maximum of n states

8) Power is same as DFA

9) Example NFA ending with zero.

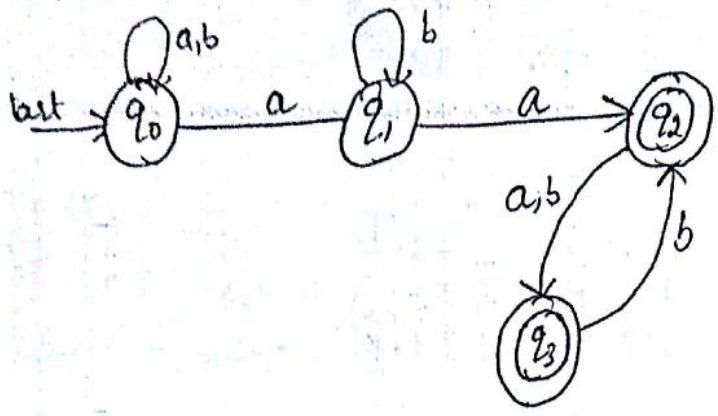


$\delta(q_0,0) = \{q_0, q_1\}$

$\delta(q_0,1) = \{q_0\}$

Conversion from NFA to DFA.

Ques! Convert the following NFA into DFA.



Sol: Let given NFA $M = (Q, \Sigma, \delta, q_0, F)$ and equivalent DFA $M_1 = (Q_1, \Sigma, \delta_1, [q_0], F_1)$, where

$Q = \{q_0, q_1, q_2, *q_3\}$, $\Sigma = \{a, b\}$, q_0 is starting state, $F = \{q_3\}$ & δ is defined as.

$\delta \mid \Sigma$	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\{q_1\}$
$*q_2$	$\{*q_3\}$	$\{*q_3\}$
$*q_3$	—	$\{*q_2\}$

δ_1 is defined as follows:

1. Keep the first row of NFA as it is with square brackets as follows:

δ/Σ	a	b
$\rightarrow[q_0]$	$[q_0, q_1]$	$[q_0]$

Now, we have two states : $[q_0, q_1]$; $[q_0]$. We select the one next state that is not a present state till now and define the transition for it. So, we have only one next state $[q_0, q_1]$, which is not a present state so far.

δ/Σ	a	b
$\rightarrow[q_0]$	$[q_0, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$

Since, $\delta_1([q_0, q_1], a) = [\delta(q_0, a) \cup \delta(q_1, a)]$

$[\{q_0, q_1\} \cup \{q_2\}] = [q_0, q_1, q_2]$ (New state)

$\delta_1([q_0, q_1], b) = [\delta(q_0, b) \cup \delta(q_1, b)]$

$[\{q_0\} \cup \{q_1\}] = [q_0, q_1]$ (Old state)

Now $[q_0, q_1, q_2]$ is the next selected state, because $[q_0, q_1]$ is defined already.

δ/Σ	a	b
$\rightarrow[q_0]$	$[q_0 q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2)$	$[q_0, q_1]$

Now the state $[q_0, q_1, q_2, q_3]$ is the next selected state (New state)

	a	b
$\rightarrow[q_0]$	$[q_0\ q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
$q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$ (New state)	$[q_0, q_1, q_3]$ ← New state)
$_0, q_1, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$ (Old state)	$[q_0, q_1, q_2, q_3]$ (old state)

Now the state $[q_0, q_1, q_3]$ is the next selected state

δ/Σ	a	b
$\rightarrow[q_0]$	$[q_0, q_1]$	$[q_0]$
$[q_0, q_1]$	(q_0, q_1, q_2)	$[q_0, q_1]$
$q_0, q_1, q_2)$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_3]$
$_0, q_1, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$ (old)	$[q_0, q_1, q_2, q_3]$ (old)
$_0, q_1, q_3]$	$[q_0, q_1, q_2, q_3]$ (old)	$[q_0, q_1, q_2]$ (old)

Let us say $q_0 \rightarrow A$

$[q_0, q_1] \rightarrow B$

$[q_0, q_1, q_2] \rightarrow C^*$

$[q_0, q_1, q_2, q_3] \rightarrow D^*$

$[q_0, q_1, q_3] \rightarrow E^*$

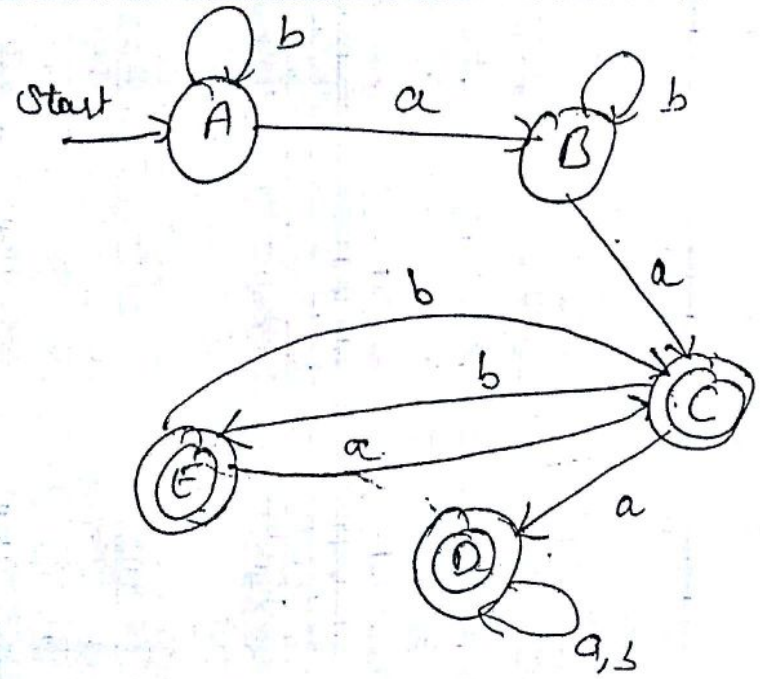
$\Rightarrow$

δ/Σ	a	b
$\rightarrow A$	B	A
B	C	B
$^*C$	D	E
$^*D$	D	D
$^*E$	D	E

∴ $Q_1 = \{[A], [B], [C], [D], [E]\}$

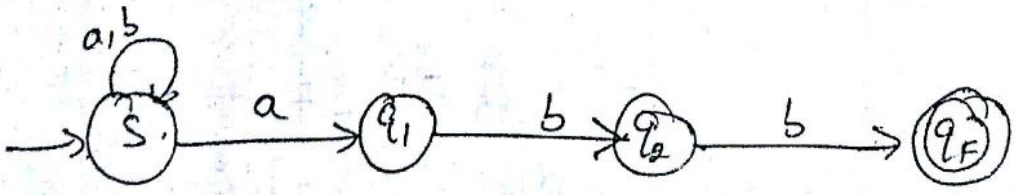
$F_1 = \{[C], [D], [E]\}$

∴ required DFA is



(DFA equivalent to NFA) Ans

Example 2. Given NFA is



find equivalent DFA.

Sol: $M = (Q, \Sigma, \delta, S_0, F)$ & equivalent DFA $M_1 = (Q_1, \Sigma, \delta_1, [S], F_1)$, where $Q = (S, q_1, q_2, q_F)$, $S_0 = \{1, 3\}$

& the starting state, $f = \{q_1\}$ & δ is defined as. (12)

δ/Σ	a	b
$\rightarrow s$	$\{s, q_1\}$	$\{s\}$
q_1	—	$\{q_2\}$
q_2	—	$\{q_f\}$
$*q_f$	—	—

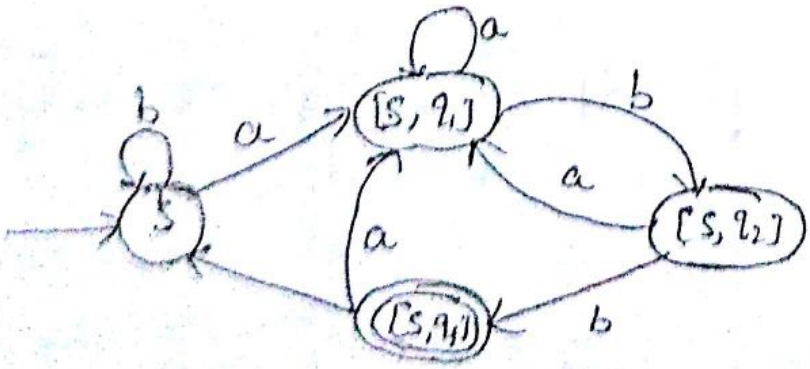
δ_1 is defined as follows:

δ/Σ	a	b
$\rightarrow$ [s]	[s, q_1]	[s]
[s, q_1]	[s, q_1]	[s, q_2]
[s, q_2]	[s, q_1]	[s, q_f]
* [s, q_f]	[s, q_1]	[s]

Now we have.

$Q_1 = \{[s], [s, q_1], [s, q_2], [s, q_f]\}$ &

$f_1 = \{[s, q_f]\}$



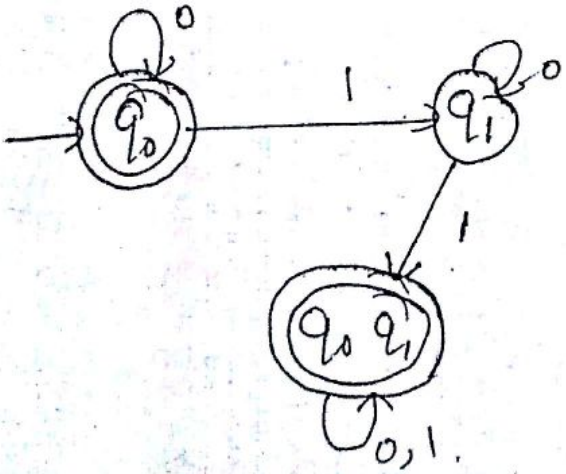
DFA equivalent to NFA) Ans

Example 3 Given NFA is

States	Inputs	
	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_1	$q_0 q_1$

Ans Required DFA is

δ / Σ	0	1
$[\rightarrow q_0]$	$[q_0]$	$[q_1]$
$[q_1]$	$[q_1]$	$[q_0 q_1]$
$[q_0 q_1]$	$[q_0 q_1]$	$[q_0 q_1]$



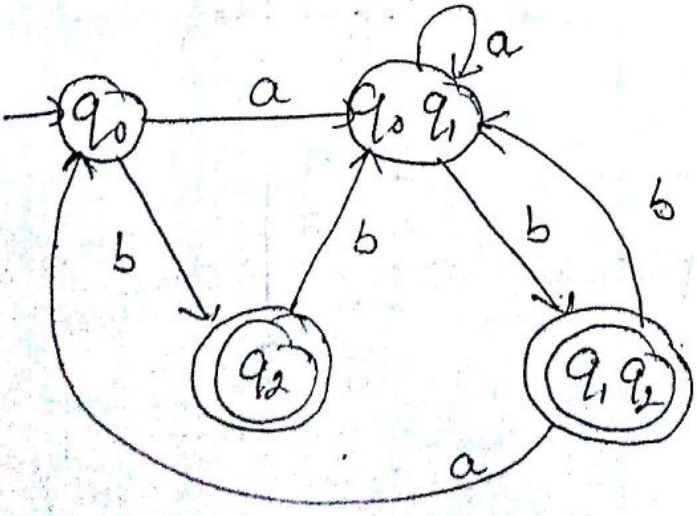
Answer

Example: Given NFA is:

State	Input a	Input b
$\rightarrow q_0$	$q_0\, q_1$	q_2
q_1	q_0	q_1
(q_2)	–	$q_0\, q_1$

Answer: Required DFA.

State	Input a	Input b
$[\rightarrow q_0]$	$[q_0\, q_1]$	$[q_2]$
$[q_0\, q_1]$	$[q_0\, q_1]$	$[q_1\, q_2]$
$[(q_2)]$	–	$[q_0\, q_1]$
$[(q_1\, q_2)]$	$[q_0]$	$[q_0\, q_1]$



Answer

Imp: Example

Give NFA is

Convert into equivalent DFA.

Ans: Transition Table for given NFA is:

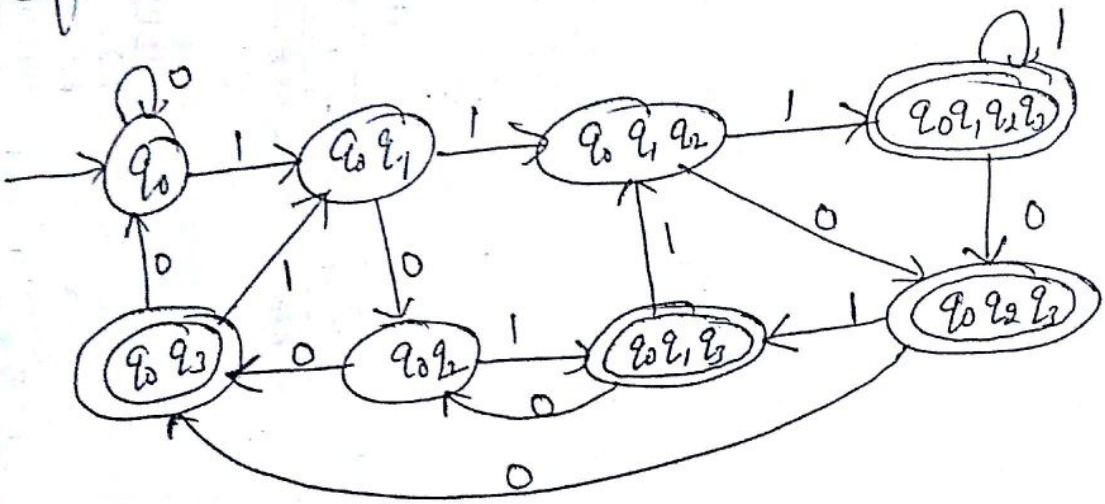
δ / Σ	0	1
$\rightarrow q_0$	q_0	$q_0 q_1$
q_1	q_2	q_2
q_2	q_3	q_3
(q_3)	–	–

Required DFA is

δ / Σ	0	1
$[\rightarrow q_0]$	$[q_0]$	$[q_0 q_1]$
$[q_0 q_1]$	$[q_0 q_2]$	$[q_0 q_1 q_2]$
$[q_0 q_2]$	$[q_0 q_3]$	$[q_0 q_1 q_3]$
$[q_0 q_1 q_2]$	$[q_0 q_2 q_3]$	$[q_0 q_1 q_2 q_3]$
$[(q_0 q_3)]$	$[q_0]$	$[q_0 q_1]$

$[q_0 q_1 q_3]$	$[q_0 q_2]$	$[q_0 q_1 q_2]$
$[q_0 q_2 q_3]$	$[q_0 q_3]$	$[q_0 q_1 q_3]$
$[q_0 q_1 q_2 q_3]$	$[q_0 q_2 q_3]$	$[q_0 q_1 q_2 q_3]$

Required DFA is:



Answer

PREPAIRED BY - DEEPAK SINGLA (A.P in CSE PIET)

Theorem : for any NFA $M = (Q, \Sigma, q_0, A, \delta)$ accepting a language $L \subseteq \Sigma^*$, there is an FA $M_1 = (Q_1, \Sigma, q_1, A_1, \delta_1)$ that also accept L.

Proof: M_1 is defined as :

1) $Q_1 = 2^Q$

2) $\Sigma_1 = \Sigma = \{a, b\}$

3) $q_1 = \{q_0\}$

for $q \in Q_1$ and $a \in \Sigma$, $\delta_1(q, a) = \bigcup_{r \in q} \delta(r, a)$

4) $A_1 = \{q \in Q_1 \mid q \cap A \neq \phi\}$

The fact that M_1 accepts the same language as M follows from the fact that for any $x \in \Sigma^*$

$$\delta_1^*(q_1, x) = \delta^*(q_0, x)$$

which we now prove using structural induction on x:

1) If $x = \wedge$ then $\delta_1^*(q_1, x) = \delta_1^*(q_1, \wedge)$
$= q_1$
$= \{q_0\}$
$= \delta^*(q_0, \wedge) = \delta^*(q_0, x)$

The induction hypothesis is that x is a string satisfying:

$$\delta_1^*(q_1, x) = \delta^*(q_0, x)$$

and we wish to prove that for any $a \in \Sigma$,

$$\delta_1^*(q_1, xa) = \delta^*(q_0, xa)$$

$$\delta_1^*(q_1, xa) = \delta_1(\delta_1^*(q_1, x), a)$$
$$\delta_1(\delta^*(q_0, x), a)$$

$$= \bigcup_{r \in \delta^*(q_0, x)} (r, a)$$

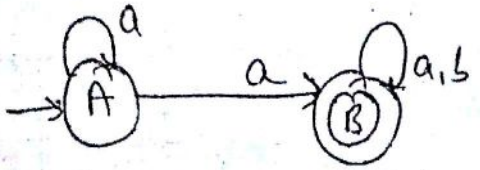
$$\boxed{\delta_1^*(q_1, xa) = \delta^*(q_0, xa)}$$

∴ M and M_1 recognise the same language.

→ A string x is accepted by M_1 if $\delta_1^*(q_1, x) \in A_1$, we can say that this is true if and only if.

$$\delta^*(q_0, x)$$

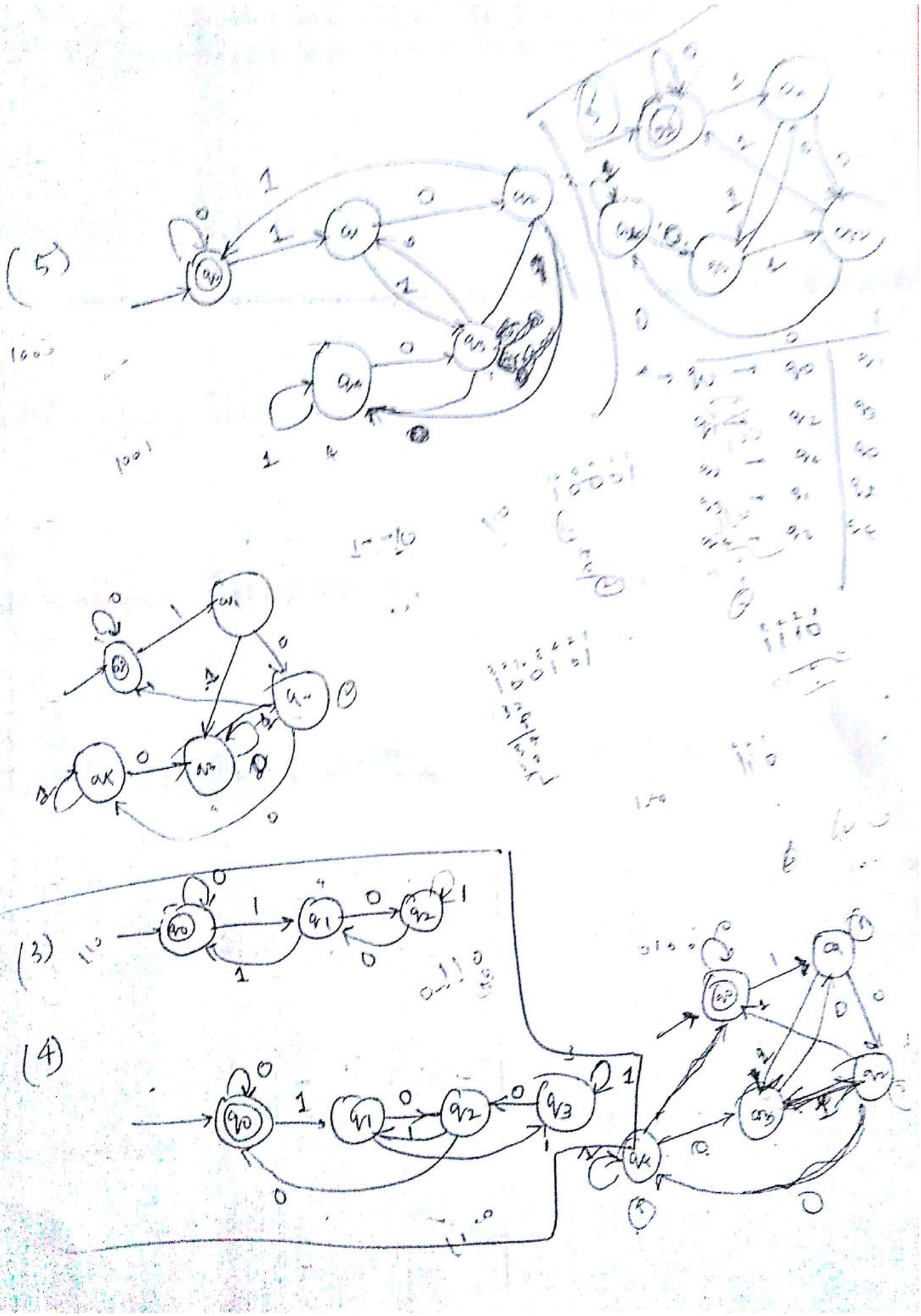
For example:



NFA

1) $Q_1 = 2^Q = \{\{A\}, \{B\}, \{AB\}, \phi\}$
2) $\Sigma = (a, b)$
3) $q_1 = \{A\}$
4) ...

(5)

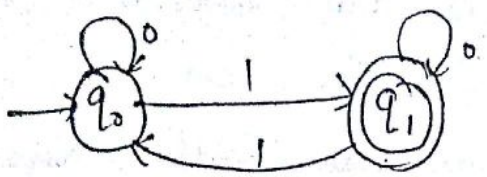


(3)

(4)

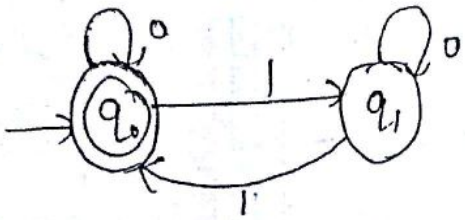
iii)- Draw an FA for odd number of 1's over $\Sigma = (0,1)$

Ans:



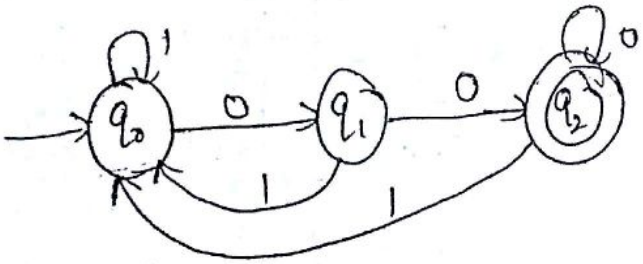
iv) Draw an FA for even number of 1's over $\Sigma=(0,1)$

Ans =



v) Draw an FA over $\Sigma = (0,1)$ which ends with 00.

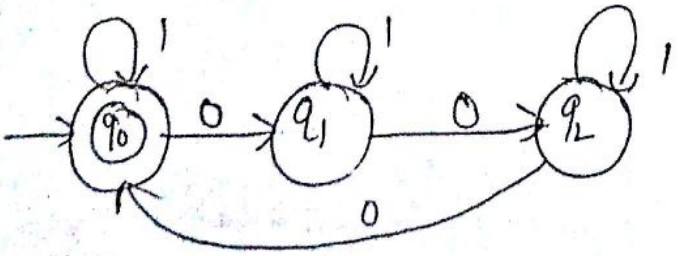
Ans =



reject → 1010

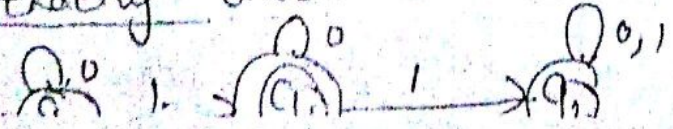
vi) Construct a FA over $\Sigma = (0,1)$ in which number of 0's in every string is multiple of 3.

Ans:



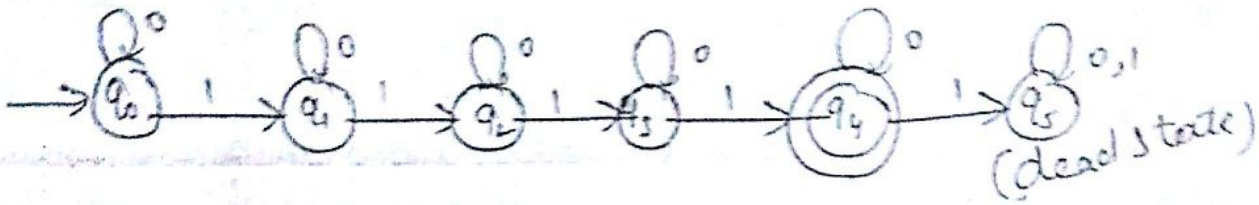
vii) Draw FA over $\Sigma = (0,1)$ which contains 1 Exactly once.

Ans



ex: Draw a FA over $\Sigma = \{0,1\}$ which accepts the strings that contain exactly four 1's.

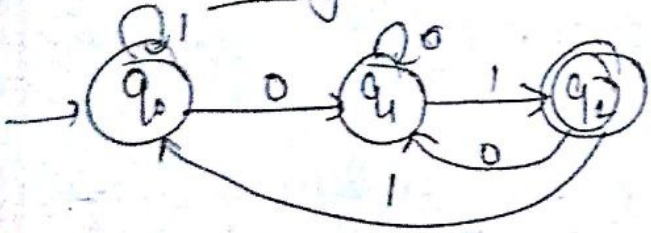
Ans:



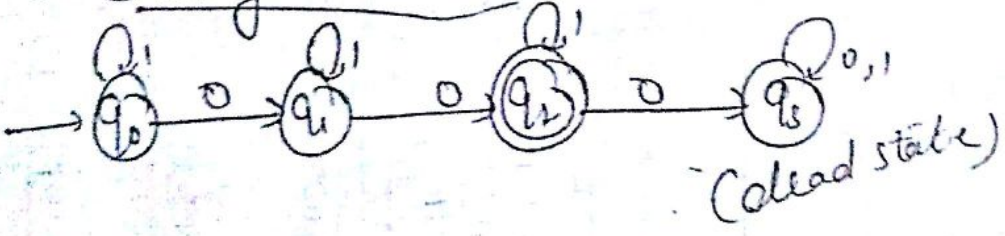
ex: Draw FA over $\Sigma = \{0,1\}$ for all strings

a) with 01 as substrings

b) Ending with 01



c) Exactly two 0's.

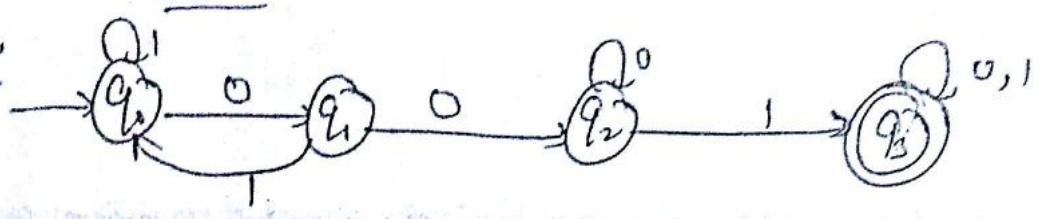


d) atleast Two 0's.

e) atmost

Qu: Draw FA over $\Sigma = (0,1)$ with substring (
001

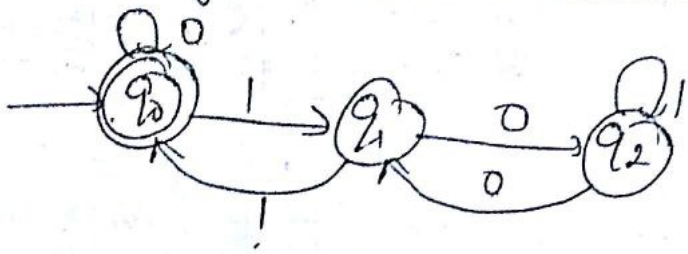
Ans:



Qu: Draw FA over $\Sigma = (0,1)$ which will accept binary number divisible by 3.

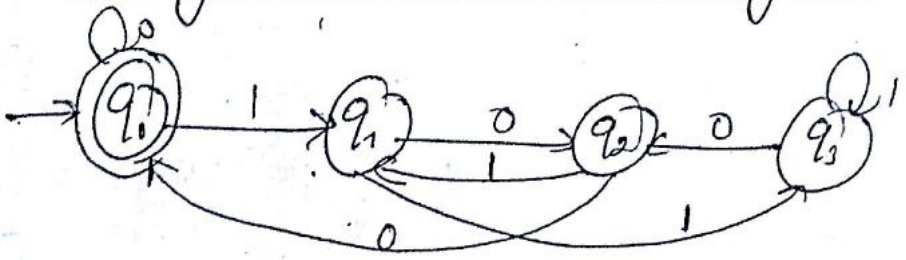
011
110
1001
1100

Ans:



Qu: Draw FA over $\Sigma = (0,1)$ which will accept binary number divisible by 4.

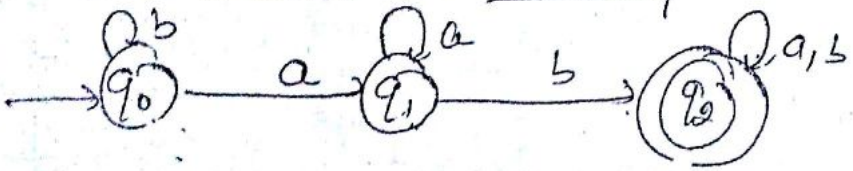
Ans:



000
1000
1100

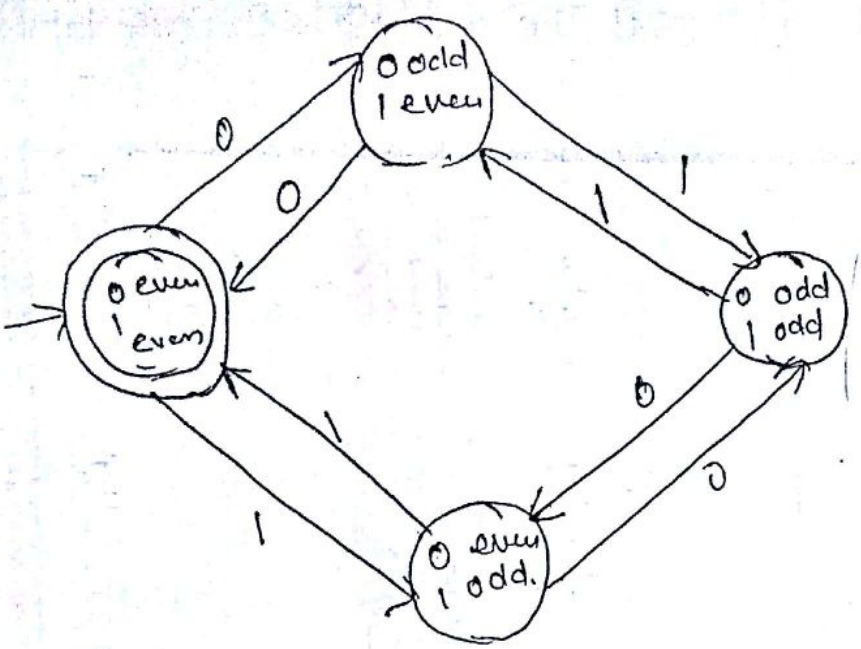
Qu: Draw FA over $\Sigma = (a,b)$

a) with ab as substring

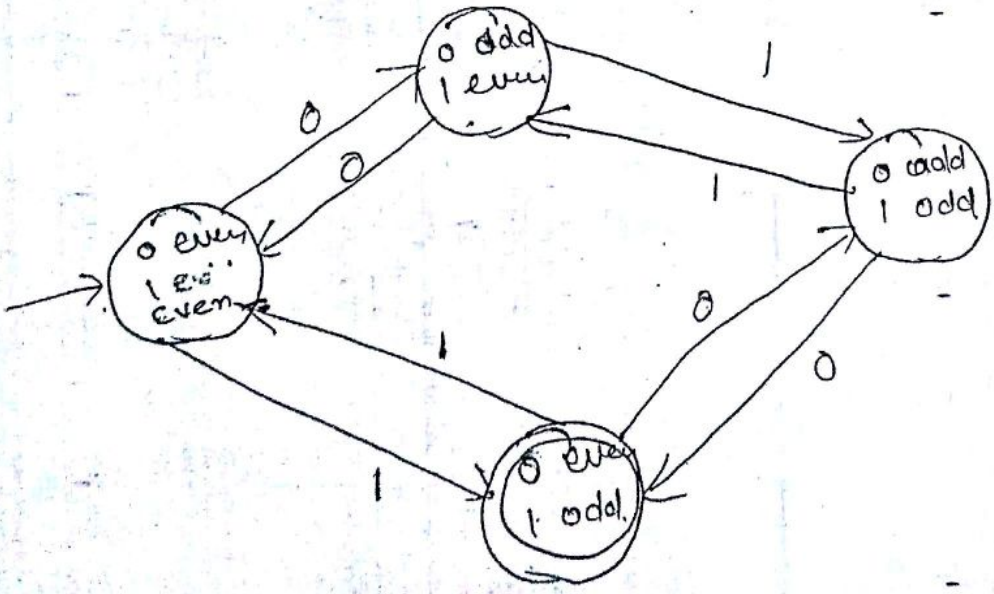


Q. Draw FA over $\Sigma=(0,1)$ which accepts

) even number of 0's and even number of 1's.



) even number of 0's and odd number of 1's.

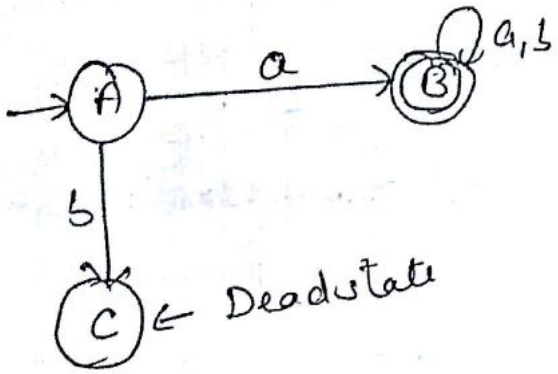


Example: Construct a DFA over $\Sigma = \{a, b\}$ which accept a string that:

1) Start with 'a'
2) Containing 'a'
3) Ending with 'a'

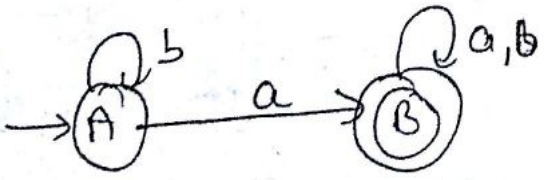
Ans: 1) Start with 'a'

∴ $L_1 = \{a, aa, aaa, ab, abb \cdots\}$

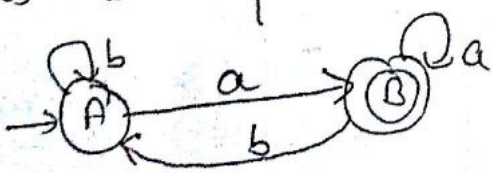


2) Containing 'a'

∴ $L_2 = \{a, aa, aba, bab, bba \cdots\}$



3) Ending with 'a'



Example: Construct a DFA which accept a string over $\Sigma = \{a, b\}$, such that the

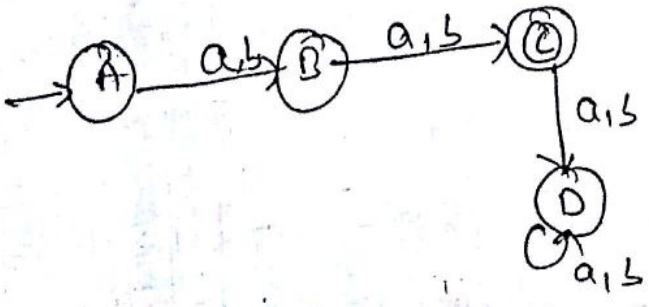
1) length of string is equal to 2
2) length of string is greater than equal 2
3) length of string is less than equal 2.

Ans:

(1) $w = \{a, b\}$, $|w| = 2$

$L = \{$ Set of all string over a, b such that the length of string is exactly $2\}$.

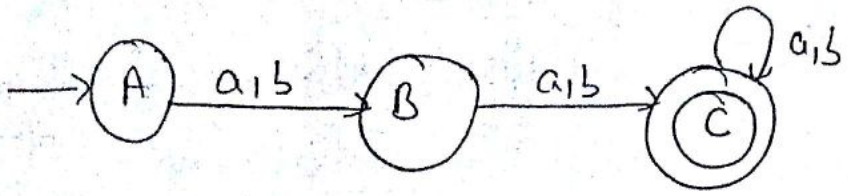
$\therefore L = \{aa, ab, ba, bb\}$



2) $w = \{a, b\}$, $|w| \geq 2$

$L = \{$ set of all string over $\{a, b\}$ such that the length of string is at least $2\}$

$\therefore L = \{aa, ab, ba, bb, aaa, aba \cdots$

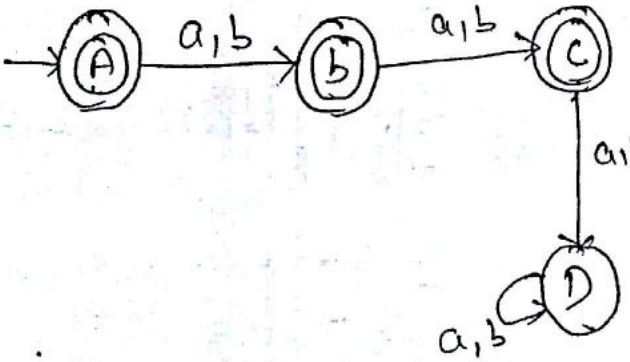


3) $w = \{a,b\}$, $|w| \leq 2$

$L = \{$ Set of all string over $\{a,b\}$ such that the length of string is atmost '2' $\}$.

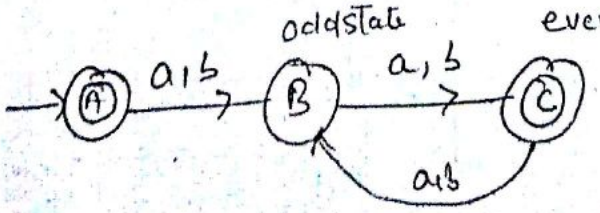
$\therefore L = \{ \epsilon, a, b, aa, ab, ba, bb \}$

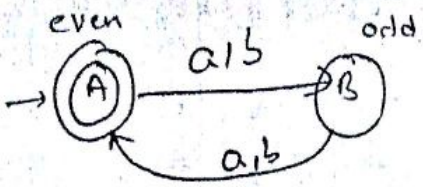
{* ϵ is always accepted at initial state in DFA}.



Example: Construct DFA, $w \in \{a,b\}$ such that $|w| \bmod 2 = 0$

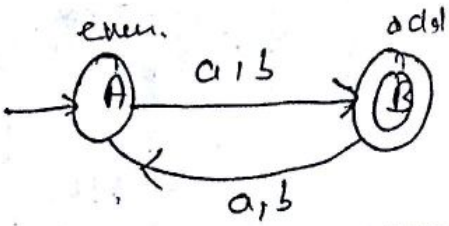
Ans: $L = \{ \epsilon, aa, ab, ba, bb, aaaa, abab \cdots \}$





Example: $w = \{a, b\}$, $|w| \bmod 2 = 1$

Ans: $L = \{$ accept odd length string $\}$.



---

*Note:- When some string length is divisible by any number then we use only number of states is equal to the number by which that string length is divided. f.e in above the string length is divided by '2'. ∴ we use '2' states in the construction of DFA.

Example: $w = \{a, b\}$, $|w| \bmod 3 = 0$

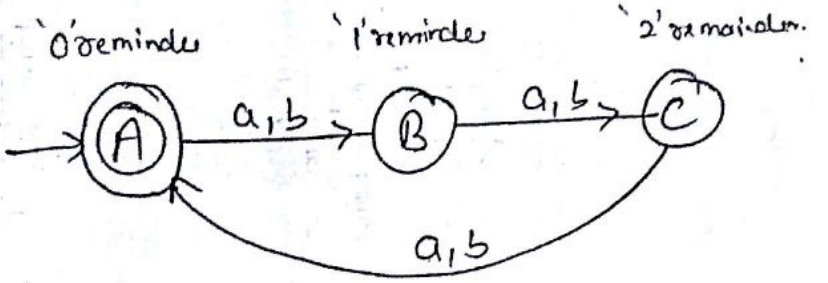
Ans: $L = \{$ accept all string length when divided by 3, the remainder is zero $\}$

* When any length string is divided by 3

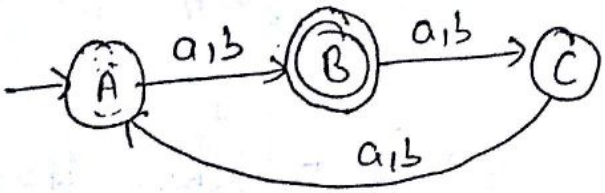
Then only remainder is 0, 1, 2.

∴ we use '3' states in the construction of DFA, as;

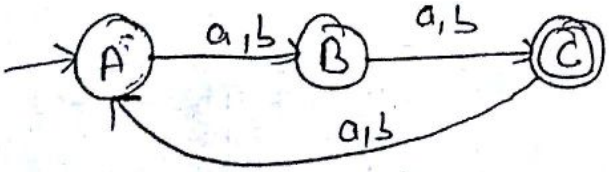
L = {ε, aaa, aba, bbb, bab, aaaaaaa, baabba ...}



Example $w = \{a,b\}$, $|w| \bmod 3 = 1$

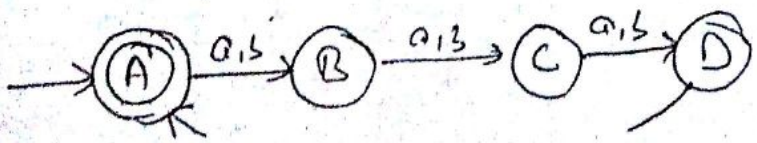


Example $w = \{a,b\}$, $|w| \bmod 3 = 2$



Example $w = \{a,b\}$, $|w| \bmod 4 = 0$

Ans: As, number is divided by '4' ∴ 0, 1, 2, 3 would be the remainders ∴

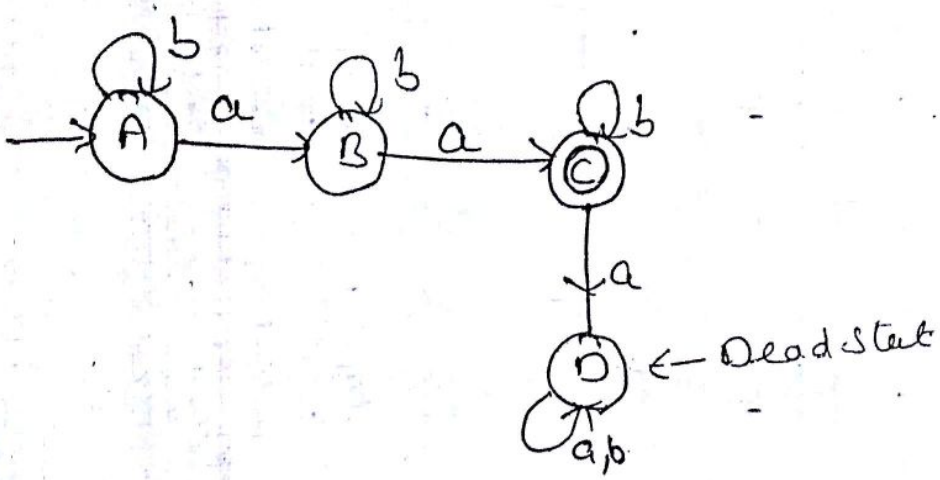


Answer

Example :- Construct DFA, $w \in \{a,b\}^*$ such that $n_a(w) = 2$

Ans :- L = {Set of all string in which the no. of 'a's is equal to 2 only}.

∴ L = {aa, aba, abba, baa, ---}.



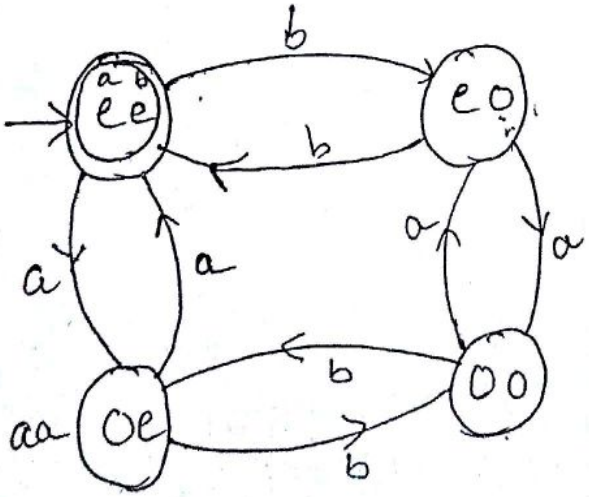
Example: Construct DFA in $w \in \{a,b\}^*$ such that

$$n_a(w) \cong 0 \bmod 2$$
$$\&\&$$
$$n_b(w) \cong 0 \bmod 2$$

Ans: $L = \{ \epsilon, aa, bb, aabb, abab, baba, bbbb \ldots \}$

$n_a(w)$	$n_b(w)$
e	e — $\{\epsilon, aa, bb \ldots\}$
e	o — $\{aab \ldots\}$
o	e — $\{aaabb \ldots\}$
o	o — $\{ab\}$

∴ classify DFA with 4 states as:



→ on seen 'a' at 'ee', it represent odd no. of 'a' & even num of 'b's as $a^{odd}\epsilon^{even}$

→ on seeing 'b' at (ee) state, it moves to because its show odd no' of b's & even no. of 'a' as $\in b$

even no. of a's, odd b's.

→ Now check {a, b} at (oe) state. check how to reach (oe), we reach (oe) by (ee) by giving input 'a' ∴ check for 'a' input at (oe).

→ $\underbrace{a\,a}_{\text{even a's}}\in$ ↑ even b's, move to (ee) state with input 'a' at (oe).

→ on giving 'b' at (oe), $a\,b$ (odd a's, odd b's), ∴ move to state (oo) with input 'b' at (oe).

→ Similar above, complet all the state with input {a, b}.

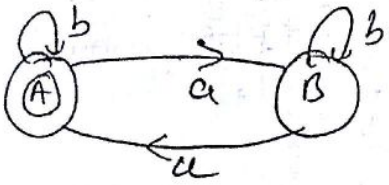
2nd Methode to Solve problem:

Ques: Construct DFA in $w \in \{a,b\}^*$ such that

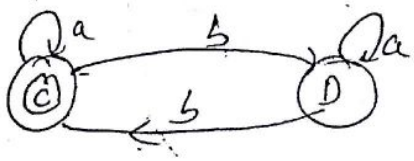
$$n_a(w) \cong 0 \bmod 2$$
$$\&\&$$
$$n_b(w) \cong 0 \bmod 2$$

Ans: Cross product Methode:

DFA for even no. of 'a'



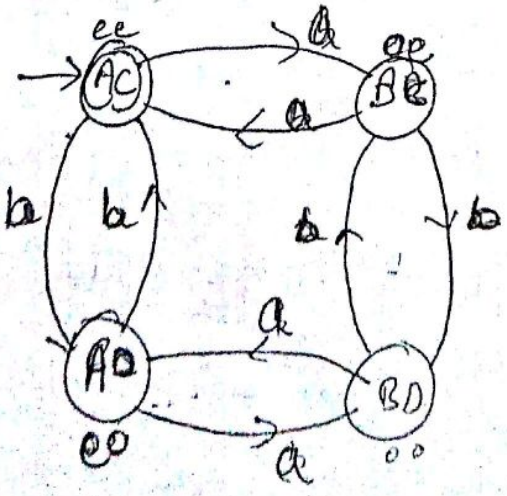
DFA for even no. of 'b's



→ Take the cross product of these two dfa's

$$\{A,B\} \times \{C,D\} = \{AC, AD, BC, BD\}$$
$$\{AC, BC, AD, BD\}$$

→ ∴ four states forms as (AC), (BC), (AD), (BD).



Check:-
AC state for a & b input

$A \xrightarrow{a} B$
$C \xrightarrow{a} C$ = (BC)

$A \xrightarrow{b} A$
$\xrightarrow{b} D$ = (AD)

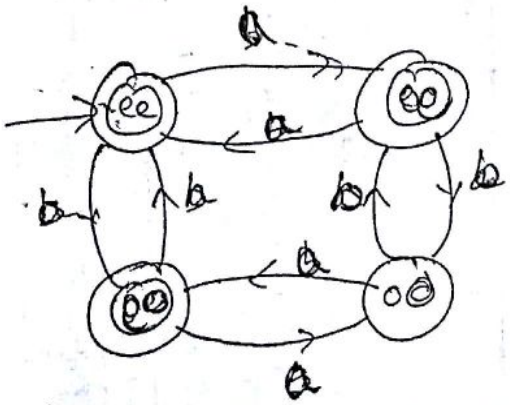
Example ⇒ Construct DFA in $w \in \{a, b\}$, such that

$$n_a(w) \simeq 0 \bmod 2$$

(OR)

$$n_b(w) \simeq 0 \bmod 2$$

Ans ⇒ Similar to last one, just make the more than one final states as {ee, eo, oe}



Ans

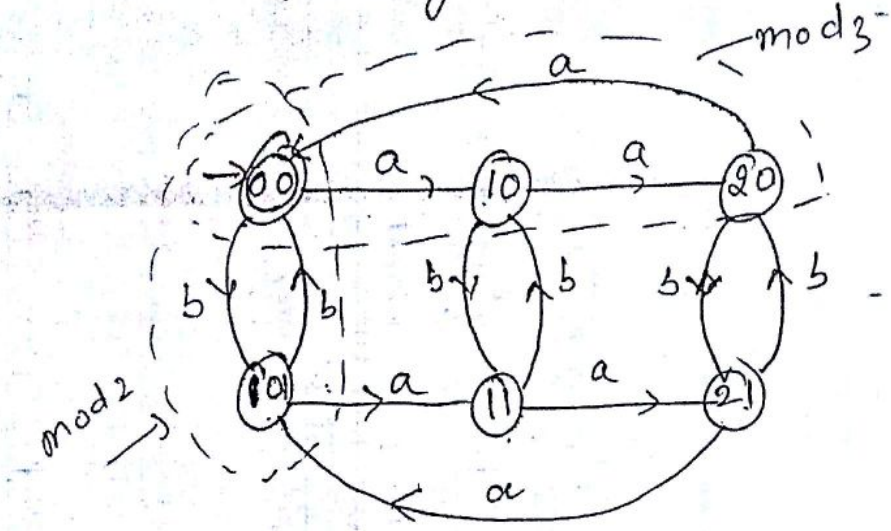
Example Construct a DFA which accepts set of all strings over {a, b} in which number of a's are divisible by 3 and no. of b's are divisible by 2.

Ans ⇒ $L = n_a(w) \simeq 0 \bmod 3$

$n_b(w) \simeq 0 \bmod 2$

No. of states = $3 \times 2 = 6$ states

→ Now count no. of 'a' horizontaly & no. of b's vertically

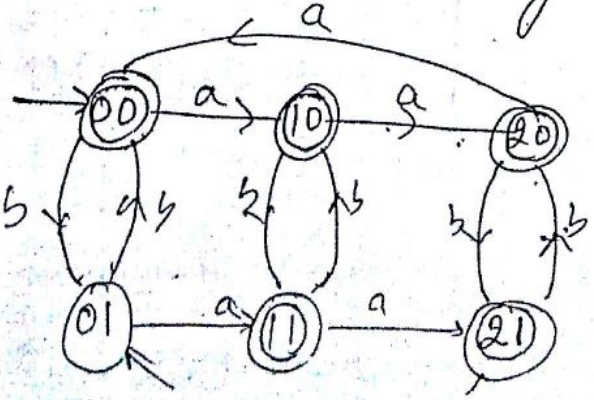


00 → 0 no of a's & 0's no. of b's.
10 → 1 " " & 0 " "
20 → 2 " " & 0 " "
01 → 0 " " & 1 " "
11 → 1 " " & 1 " "
21 → 2 " " & 1 " " b.

Example: Construct DFA, $w = \{a,b\}^*$ such that

$$n_a(w) \bmod 3 \geq n_b(w) \bmod 2$$

Ans: L = { no. of 'a' are greater than equal no. of b's }



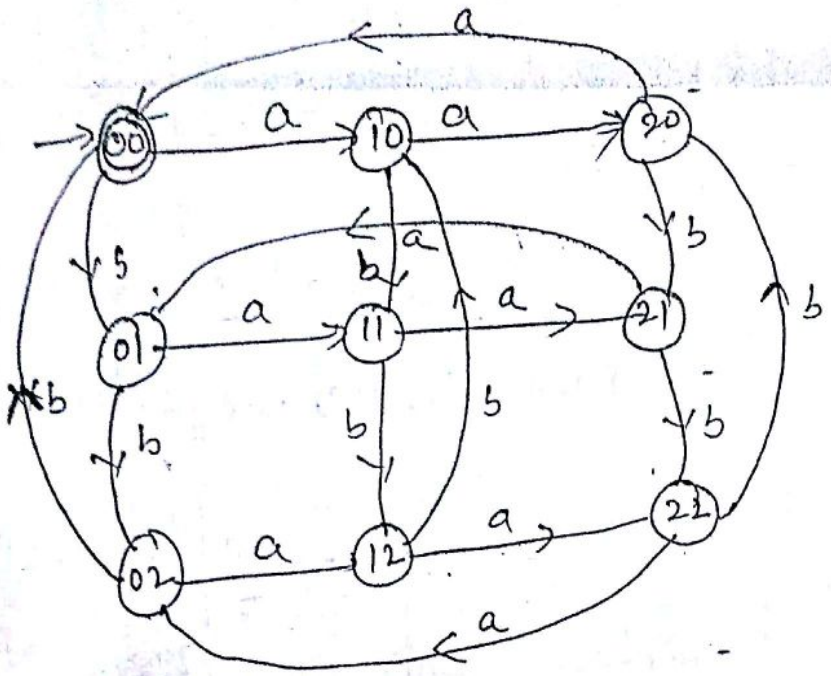
F = {10, 20, 21, 00, 11}

Then all are final states bcz in this no's o's are greater or equal to no. of b's.

Example: Construct DFA, $w \in \{a,b\}^*$, such that

$n_a(w) \equiv 0 \bmod 3$ &&
$n_a(w) \equiv 0 \bmod 3$.

Ans:



* If $n_a(w) \bmod 3 = 1$ & $n_b(w) \bmod 3 = 2$ then make {12} state as final state.

* If $n_a(w) \bmod 3 > n_a(w) \bmod 3$, then the final states are {10, 20, 21}.

or

$(n_a(w) > n_b(w)) \bmod 3$ same as above.

---

*Note:- If the statment is $n_a(w) \equiv 0 \bmod n$ & $n_a(w) \equiv 0 \bmod m$

then the total no. of states is $n \times m$.

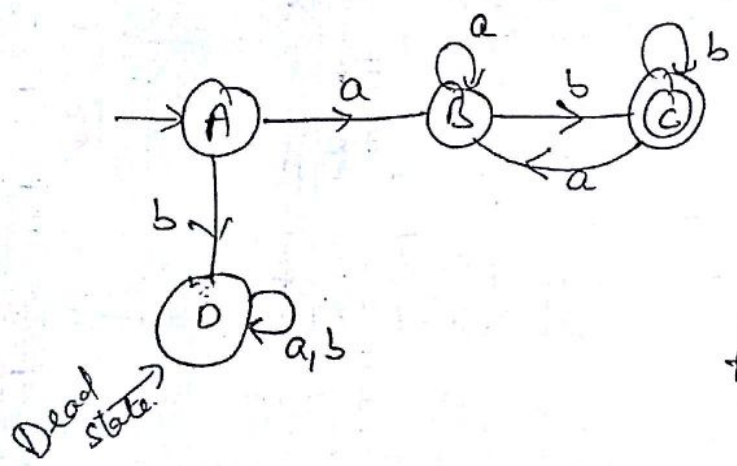
Example: Construct a DFA, $w \in \{a,b\}^*$ such that string start with a & End with b.

Ans:

$L_1 = \{ a, ab, bb, aab \cdots \}$

$L_2 = \{ b, ab, bb, aab \cdots \}$

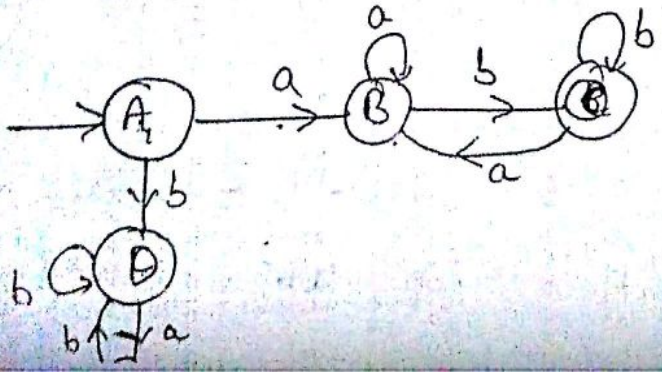
$L_1 L_2 = \{ ab, aab, abb \cdots \}$



Ans

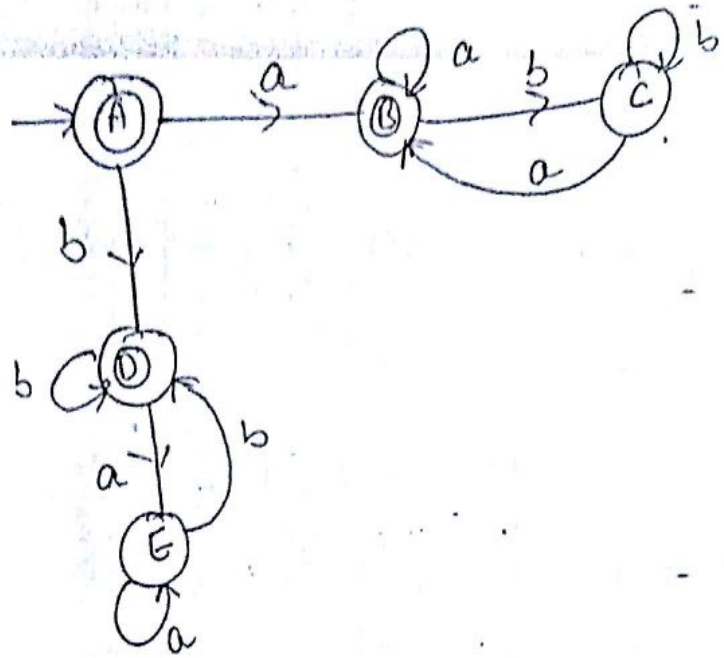
Example: Construct DFA, $w \in \{a,b\}^*$, such that string start with different symbol & End with different symbol.

Ans: $L = \{ ab, ba, aab, abb \cdots \}$



Example: Construct DFA, which accept string that (Start & End with same Symbol.

Ans: $L = \{\epsilon, a, b, aa, bb \cdots\}$.



Note:*

Previous example & this example is Complement of each others. The no. of states are same with same moves in both diagram only the Difference is that the final state in one diagram is a normal state in other & Vice-versa.

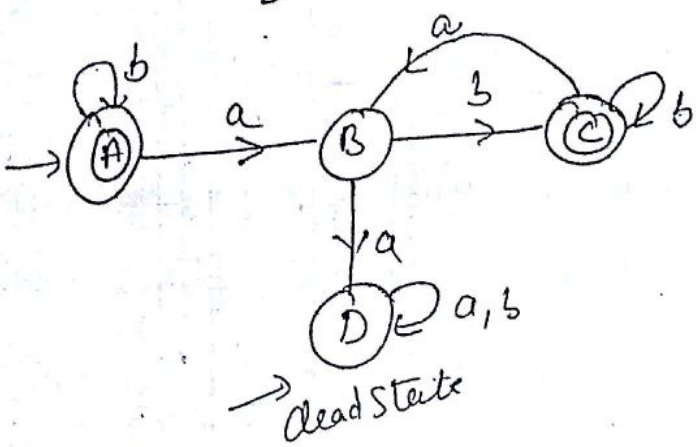
Complimentation:- i) DFA. ii) $(Q, \Sigma, \delta, q_0, F)$

$\Downarrow$

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

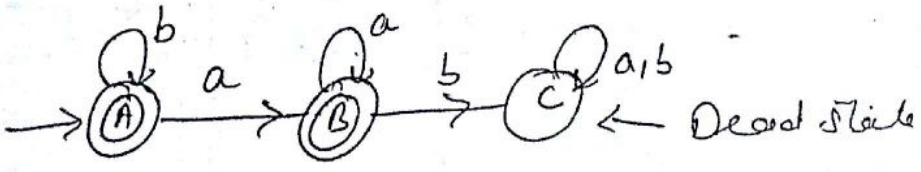
Example: Construct a DFA which accepts set of all strings over {a,b} in which every 'a' is followed by 'b'.

Ans: $L = \{\epsilon, ab, abab \text{---} b, bbb\}$.



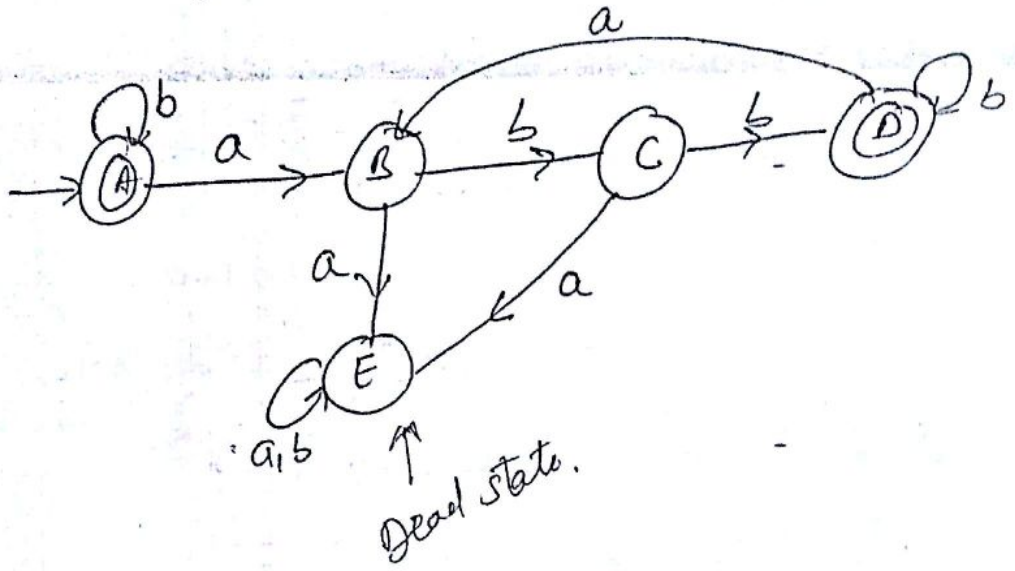
Example: Construct a DFA which accept set of all strings over {a,b} in which every 'a' should not be followed by 'b'.

Ans: $L = \{\epsilon, a, aa, aaa \text{---} b, bb, bba \text{--}\}$



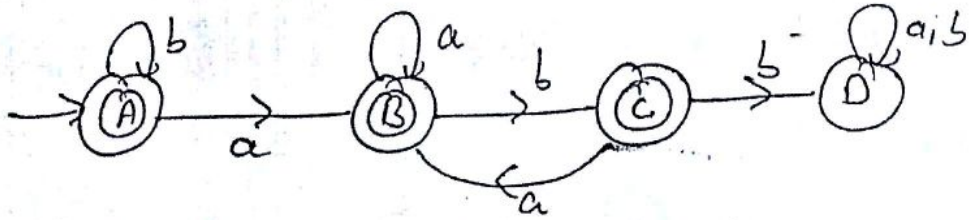
Example: Construct DFA, in which every 'a' has to be followed 'bb'.

Ans: $L = \{\epsilon, b, bbb \ldots, abb, bbabb \ldots\}$



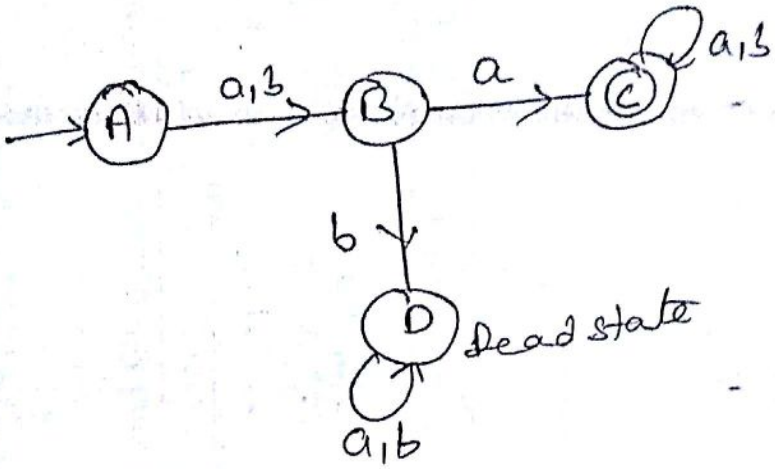
Example: Construct DFA, in which every 'a' should not followed 'bb'.

Ans: $L = \{\epsilon, b, bb, bbb \ldots a, aa, aaa, ab, aab\}$



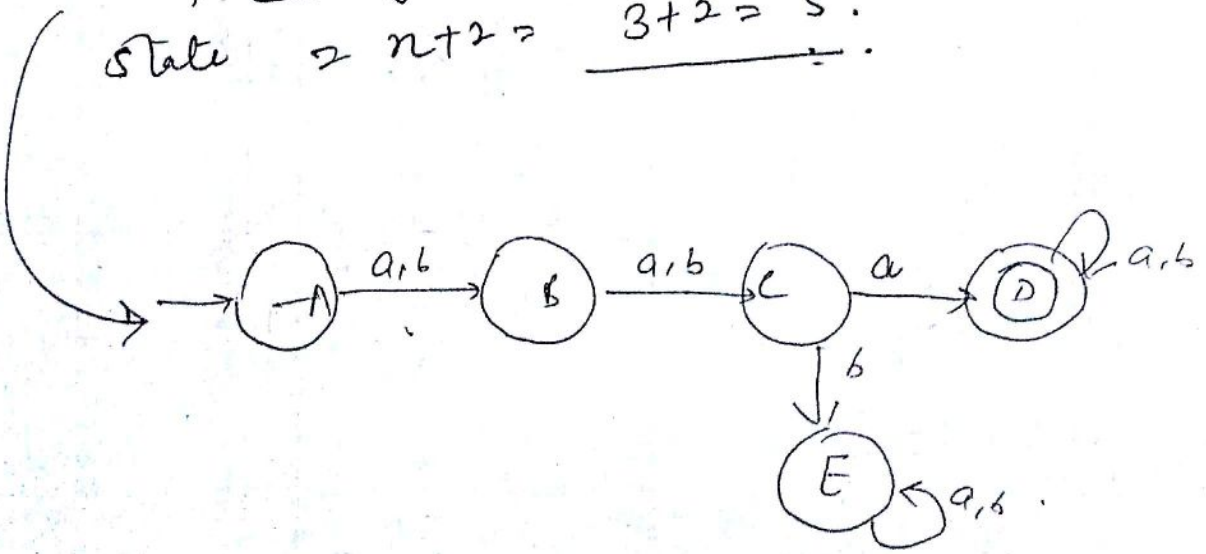
Example: Construct a DFA which accepts set of all strings over $\{a,b\}$ such that second symbol from LHS is 'a'.

Answer: $L = \{aaa, baa, baaa, aabb \ldots\}$



* If '2' symbol from LHS is recognised then total no. of states = $n+2$
  $2+2 = 4$ state

e.g. If 3rd symbol from LHS then total no. state = $n+2 = 3+2 = 5$.



Example:- Construct a DFA, which accept set of all strings over {0,1}, which when interpreted as binary number is divisible by '2'.

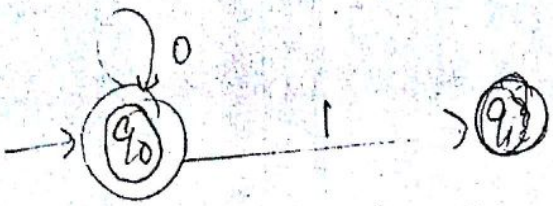
Ans:- In this problem, the number of states is equal to the no. which the number is divided. i.e in this accept a string as binary number divisible by '2'. ∴ '2' states are formed as:

Now check the 0 & 1 transition on state q_0 first else find the number of remainder form when any number is divided with 2.

The 2 remainders as 0 & 1 are formed ∴ Create 2 states as q_0 & q_1.

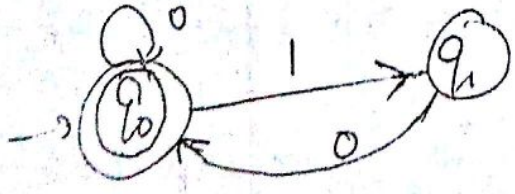
Now check for 0 transition on q_0. as when 0 is divided by 2 it generate the remainder as 0. ∴ on 0 transition it will be rema

	0	1
q_0 →	q_0	q_1
q_1 →	q_0	q_1



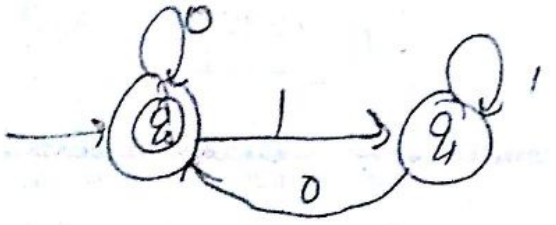
→ Now check for Transition '1' on q_0. As '1' is divided by 2 then the remainder generated as 1. so on giving '1' on q_0, it goes to state q_1 which shows the remainder as 1.

→ Now check transitions over q_1. we reached q_1 with input 1, Now if apply 0 then the Binary number is 10 which is equivalent to Decimal number 2 Now when it is divided by 2 the remainder is 0. so when apply '0' on q_1 it moves to q_0 which shows the remainder '0' state.



→ Now when apply 1 on q_1. it form the Binary number 11 which is Decimal number 3. when 3 is divided by 2, it

generate remainder as 1. so on giving 1 as input on q_1 it move to q_1 state.



→ Make a q_0 state as final state, Because in this given problem the binary number is divisible by 2 the remainder is zero. so make q_0 state as final state.

Ans

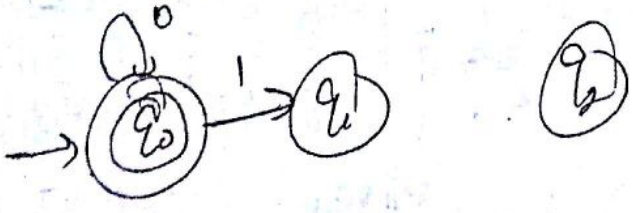
2. Construct a DFA, which accept set of all strings over $\{0,1\}$, which when interpreted as binary number is divisible by 3.

Ans As num is divisible by 3, it generate remainder as 0, 1, 2. so make 3 state as q_0, q_1, q_2 & make q_0 as final

→ Check 0,1 over q_0.

* for '0' input, when 0 is divisible by 3 it generate remainder as 0. So on 0 input it moves to q_0 itself.

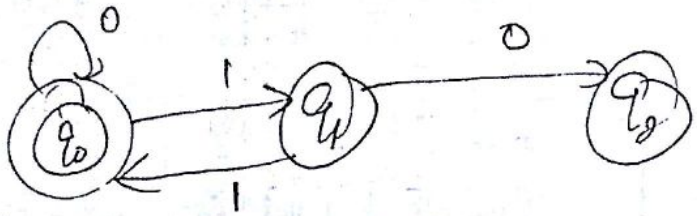
* for '1' input, when 1 is divisible by 3 it generate the remainder as '1'. So on '1' input it moves to q_1 state. as:



→ Check 0,1 over state q_1.

* for '0' input, we reach q_2 with input '1' from q_0. So the number is 10 in Binary is equivalent to '2' Decimal when divided by '3' it generate the remainder as '2'. So over '0' input it moves to state

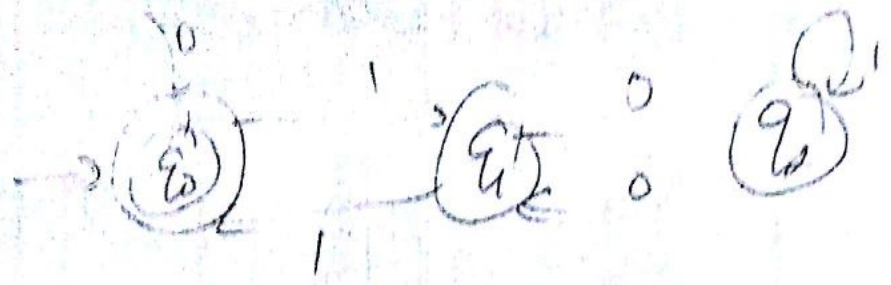
* for '1' input, the Binary number is formed as 11 which is equivalent to Decimal number '3', when it is divided by '3' it generate remainder as 0' so, as '1' input, it move to state q_0.



6. Now check transition over q_2 with input 0 & 1

* for 0, it form the Binary number as 100 which is equivalent to '4' - when divided by '3', it generate remainder as 1. so, move to state q_1.

* for 1, it form the Binary Number as 101 which is equivalent to '5', when divided by '3' it generate remainder as 2. so it move to state q_2. as shown:,



Transition Table:

	0	1
$\rightarrow *q_0$	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

Example If the above question is modified & says accept a binary string % 3 mods

Ans :- that means when any number is divide by '3' the remainder should be 1 or 2 or 0 make the q_i as final state from above. as

Ans