FORMAL LANGUAGES AND AUTOMATA THEORY

unit I - introduction to FLAT.

FA: DFA = NFA

unit II - Regular Expression (RE).

PDA: DPDA ≠ NDFDA

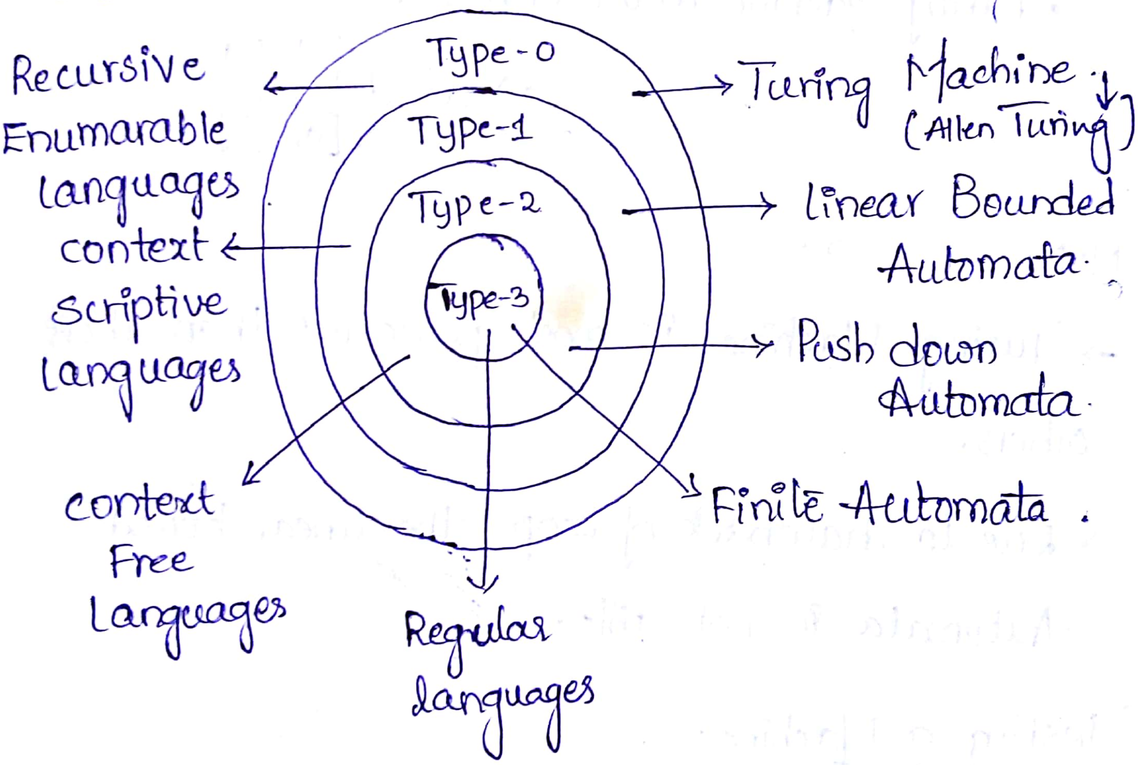
unit III - Grammar Formalism

unit IV - PUSH DOWN AUTOMATA (PDA).

TM: DTM = NDTM

unit V - Turing Machines (TM).

chomsky's classification:



(1) Finite Automata:-

$a^n, n>0$ - a, aa, aaa, aaaa, ...

drawback:-

→ doesn't have stacks. (0 stacks).

→ no storage elements

(2.) Push Down Automata:

$a^n b^n, n>0$ – ab, aabb, aaabbb, ...

- contains one stack.

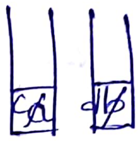
(3.) Linear Bound Automata:

$a^n b^n c^n, n>0$ – abc, aabbcc, aaabbbccc, ...

- contains two stacks.

(4.) $a^n b^n c^n d^n, n>0$ – abcd, aabbccdd, ...

- Turing Machine also contains two stacks

→ abcd 

<u>Note:</u>

→ Turing Machine is most powerful than above others.

→ Due to drawback of loops the linear Bound Automata is not able.

<u>Testing a Machine:</u>

- Taking a string which is in Machine.
- Rejecting a string which is not available in Machine.

FINITE AUTOMATA :- Mathematical representation of a machine.

→ It is a 5-Tuple Machine.

→ 5-Tuple consists of [Q, Σ, δ, S, F]

Q - set of states.

Σ - set of alphabets.

δ - Transisions (or) Grammer Rules.

S - starting state (First state)

F - Final state.

δ is defined|derived from $Q \times \Sigma \rightarrow Q$

Representations:

○ → intermediate state.

◎ → Final state

→○ → Starting state

ϵ → Epslon length of the String = 0 (starting)

$(a,b)^*$ → a, b whole star means possible strings.

Prblm-1 :

construct a finite Automata on input alphabet $\Sigma(a,b)$ having all strings formed with the given alphabet.

(1) The set of possible strings are:

$(a,b)^* - \epsilon, a, b, ab, \ldots\ldots$

(2) State diagram:

(3) $Q = (S_1)$

$\Sigma = (a,b)$

$\delta =$

set of alphabets

States	a	b
$\rightarrow$ S_1	S_1	S_1

Starting state - S_1

Final State - S_1

Note: no. of transistions in a state = no. of alphabets

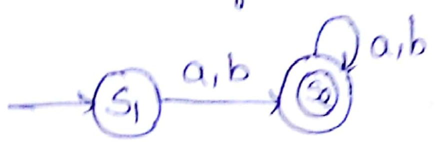
Prblm-2:-

construct a finite Automata on input symbol/alphabet $\Sigma(a,b)$ which accept all the possible strings except Epsoln (ε).

(1.) The possible strings are:

$\Sigma(a,b)^{+}$ - a, b, ab, ba, aa, bb, ------

(2.) State diagram:



(3.) $Q = (S_1, S_2)$

$\Sigma = (a, b)$

starting state - S_1

Final state - S_2

δ =

	a	b
$\rightarrow S_1$	S_2	S_2
S_2	S_2	S_2

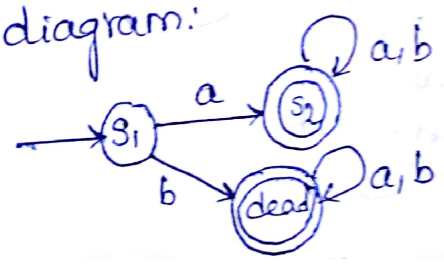
Prblm-3:

construct a finite Automata on input symbol $\Sigma(a,b)$ where each string starts with 'a'.

(1.) The possible strings are:

(a, b) - a, aa, ab, aaa, abb, aba, aab, ----

(2.) State diagram:



Here dead state → intermediate path.

$S_2 \xrightarrow{a} S_2$
$S_2 \xrightarrow{b} S_2$

- S_1 with 'b' goes to dead state.
- S_1 with 'a' goes to S_2 state.

(3.) $Q = (S_1, S_2, \text{dead})$ (or) (S_1, S_2)

$\Sigma = (a, b)$

$\delta =$

	a	b
→ S_1	S_2	dead
S_2	S_2	S_2
dead	dead	dead.

Starting state $= S_1$.
Final State $= S_2$.

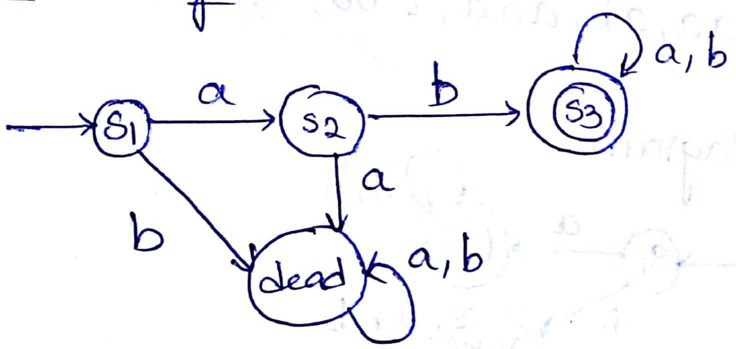
Prblm-4:

construct a finite Automata on $\Sigma(a,b)$ where each string starts with ab.

(1.) The possible strings are:

(a,b) – ab, abb, aba, abaa, abab, abbb, abba, ----

(2.) state diagram:



(3) $Q = (S_1, S_2, S_3)$

$\Sigma = (a, b)$

Starting state - S_1

Final state :- S_3

δ =

	a	b
$\rightarrow S_1$	S_2	dead
S_2	dead	S_3
S_3	S_3	S_3

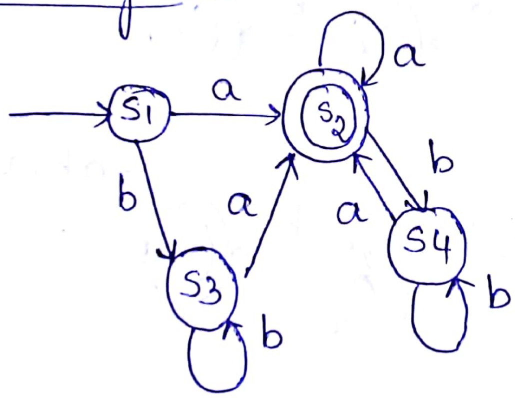
Prblm-5:

Construct a finite Automata on $\Sigma(a,b)$ where each string st ends with a.

(1.) The set of possible strings are:

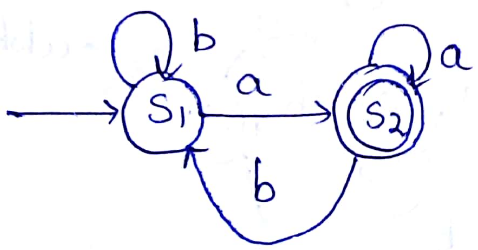
(a,b) - a, aa, ba, aaa, aba, bba, baa, aaaa, abba, abaa, - - - - -

(2.) state diagram:



(or)

Minimal Finite Automata - having low no. of states.



Scanned by CamScanner

3.) $Q = (S_1, S_2)$

$\Sigma = (a, b)$

Starting state - S_1.

Final state - S_2.

$\delta =$

	a	b
$\rightarrow S_1$	S_2	S_1
S_2	S_2	S_1

Prblm-6 :-

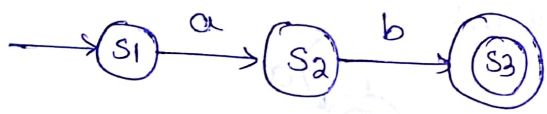
Construct a finite Automata with input alphabet $\Sigma(a,b)$ where each string ends with ab.

(1) The possible strings are :

(a, b) - ab, aab, bab, abab, bbab, aaab, baab, ...

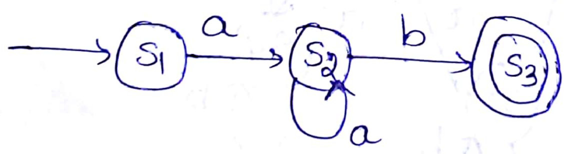
(2) State diagram :

Level-I :



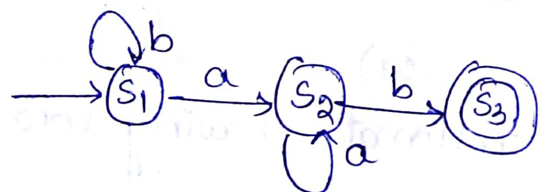
-ab

Level-II.



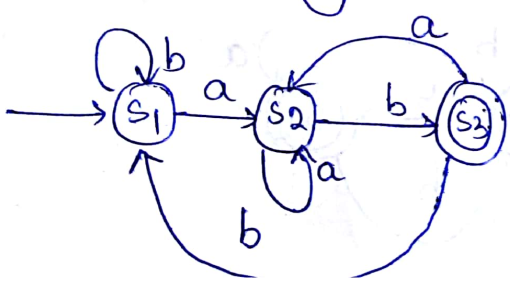
-ab, aab.

Level-III :



-ab, aab, bab

Final:



-abbab

(3.) $Q = (S_1, S_2, S_3)$

$\Sigma = (a, b)$

starting state $- S_1$

final start $- S_3$.

$\delta =$

	a	b
$\rightarrow S_1$	S_2	S_1
S_2	S_2	S_3
S_3	S_2	S_1

Prblm-7:

construct a finite Automata with input symbol (Σ (a,b)) each string starts with a and ends with b.

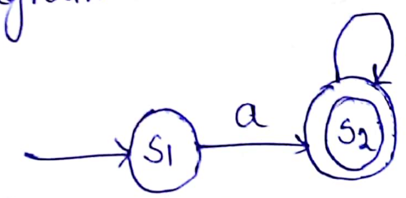
(1.) The possible strings are:

(a,b) - ab, abb, aab, abab, aaab, abbb, aabb - -

(2.) state diagram:

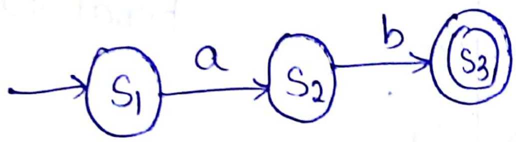
-ab

level-I:



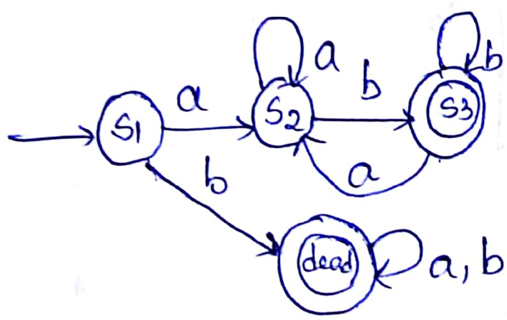
(OR)

-ab



level-II

-ab, abb



(3.) $Q = (S_1, S_2, S_3)$

$\Sigma = (a, b)$

Starting state - S_1.
Final state - S_3.

$\delta =$

	a	b
$\rightarrow S_1$	S_2	dead
S_2	S_2	S_3
S_3	S_2	S_3

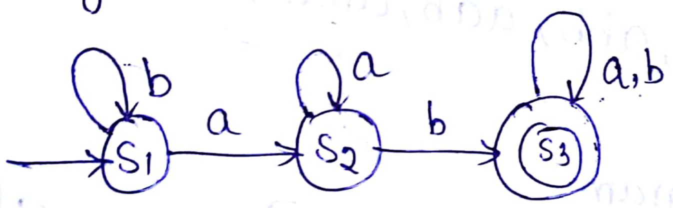
8. construct a finite Automata on input symbol $\Sigma(a,b)$ where each string having ab as a substrin

(1.) The possible strings are

(a,b) - ab, aba, bab, abb, aaab, aab, - - - - - -

(2.) The State diagram:

level-I:



(3.) $Q = (S_1, S_2, S_3)$

$\Sigma = (a, b)$

Starting state - S_1
Final state - S_3

$\delta =$

	a	b
$\rightarrow S_1$	S_2	S_1
S_2	S_2	S_3
S_3	S_3	S_3

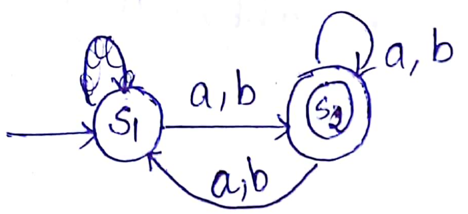
9. construct a F·A on $\Sigma(a,b)$ where each string length is equal 3.

(1) The possible strings are:

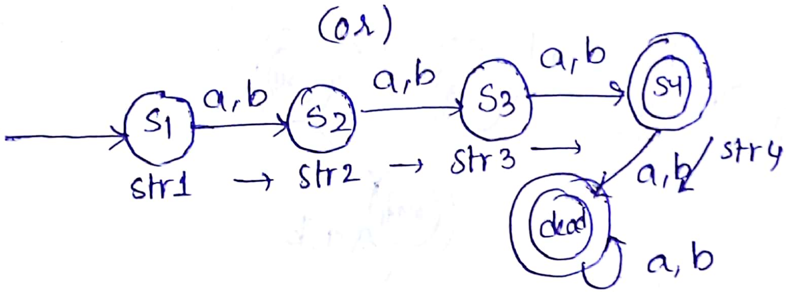
(a,b) - aaa, bbb, baa, aba, aab, aab, abb, bba

(2.) The state diagram:

level- I



(or)



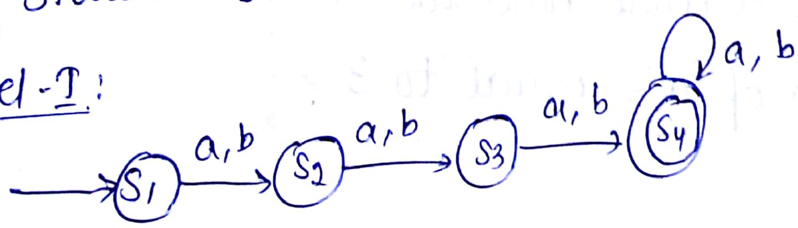
10. construct a Finite Automata on $\Sigma(a,b)$ where length of Each string is Minimum 3. can be 3, 4, 5, - - -

(1) The Minimum '3' strings possible are:

(a,b) - ~~a, ab, ba, aa, bb~~, aaa, bbb, baa, aba, aab, abb, bba, - - - .

(2) state diagram

<u>level - I</u>:



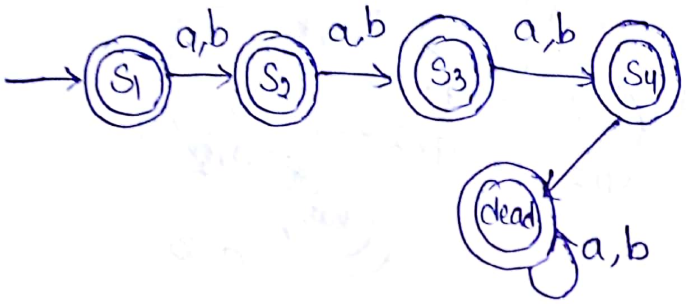
$Q = (s_1, s_2, s_3, s_4)$.

11. construct a finite Automata on input symbol (a,b) where each string length is maximum 3.

(1.) The possible strings.

(a,b) - ϵ, a, b, ab, ba, aa, bb, aaa, bbb, aab, aba, baa, abb, bab, bba, bbb.

(2.) State diagram:



(3.) $Q = (S_1, S_2, S_3, S_4)$

$\epsilon = (a, b)$

$\delta =$

	a	b
$\rightarrow S_1$	S_2	S_2
S_2	S_3	S_3
S_3	S_4	S_4
S_4	dead	dead

$S - S_1$

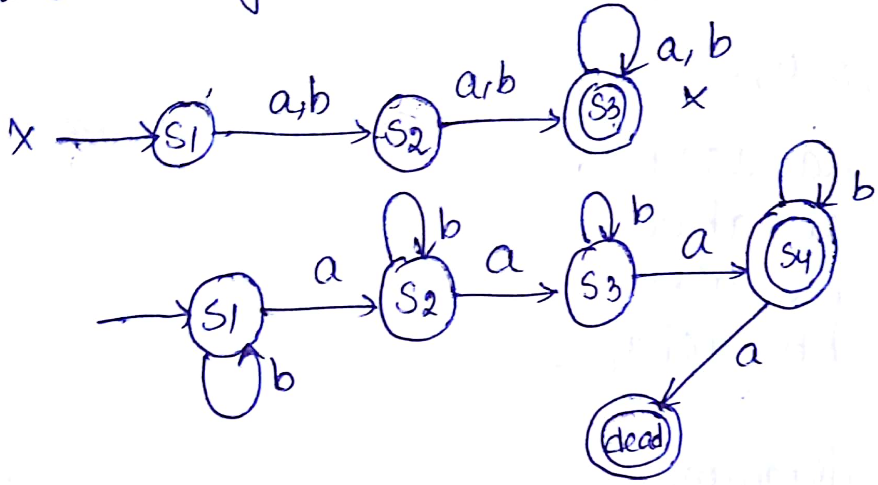
$F - S_1, S_2, S_3, S_4$.

12. construct a Finite Automata where each string having no. of 'a's equal to 3.

(1.) The possible strings are:

(a,b) - aaa, aaba, baaa, aaab, abaa, - - - -

(2.) The state diagram:



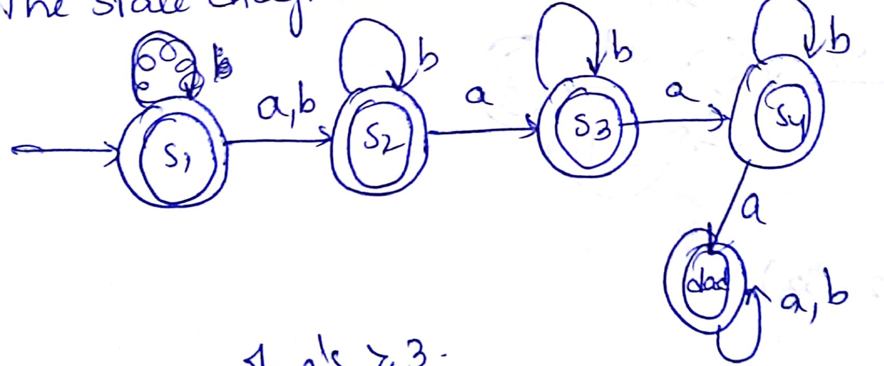
13. construct a F.A on $\Sigma(a,b)$ where each string

(1.) having no. of a's ≥ 3.

(2.) having no. of a's ≤ 3.

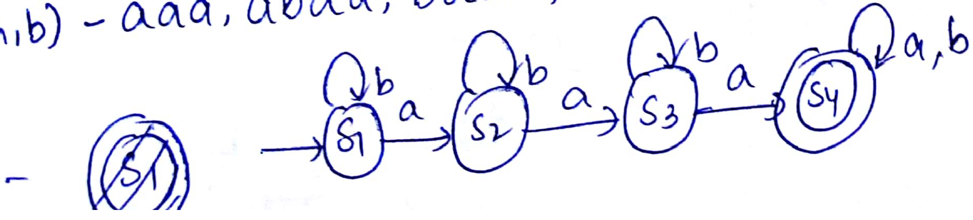
(2.) $\Sigma(a,b)$ – a, ab, ba, aa, aaa, aba, baa, aab, bba, aaab, abaa, abba – - - —

The state diagram:



(1.) having no. of a's ≥ 3.

$\Sigma(a,b)$ - aaa, abaa, baaa, aaab, aaba, - - - - -



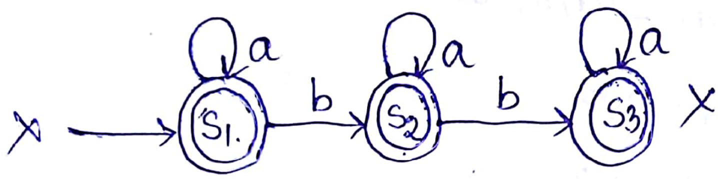
14. Construct a Finite Automata where each string length is divisible by 2. $\Sigma(a,b)$.

(1.) The possible Strings are:

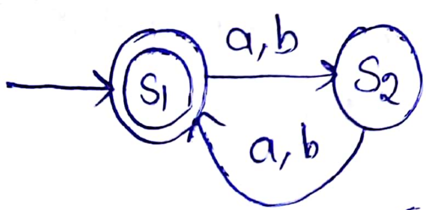
(a,b) - 0, 2, 4, 6

ε, aa, aaaa
ab, abaa
ba, abba
bb, aaba, ...

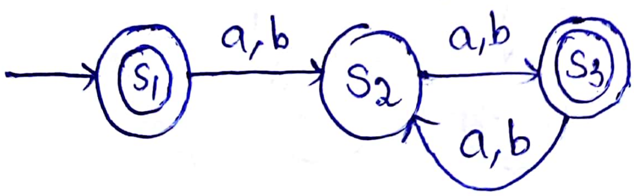
(2.) The State diagram:



Minimal Finite Automata:



(or)



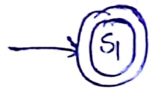
$Q = (S_1, S_2, S_3)$

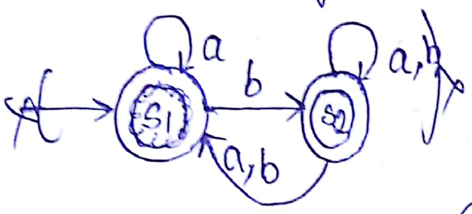
15. On Odd length Strings. (1, 3, 5, 7, ...)

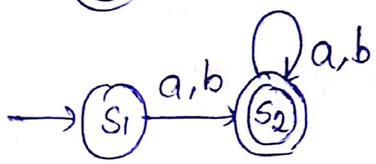
(1) The possible strings are:

(a,b) - a, aaa, aba, bba, bbb, aab, . . . abbab

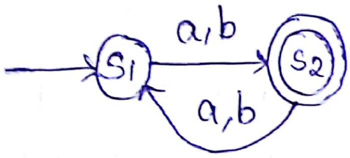
(2) The state diagram is:



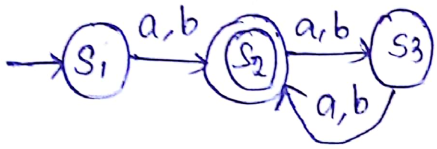




(or)



→ Minimal Finite Automata.

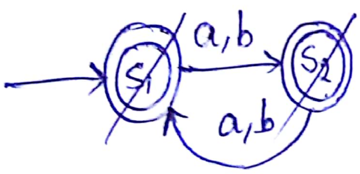


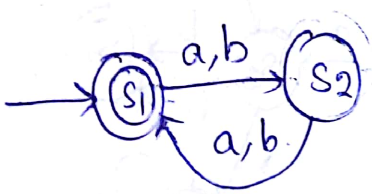
16. construct a FA on Σ(a,b)

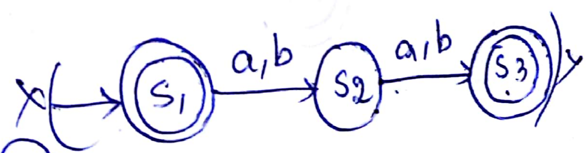
where each string having even no. of a's.

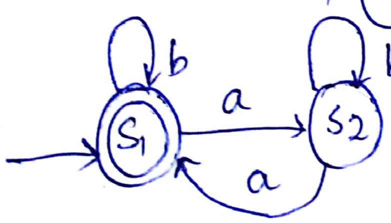
(1) The possible strings are:

(a,b) - ϵ, aa, b, aab, baa, aba, bb, bbb abaaa









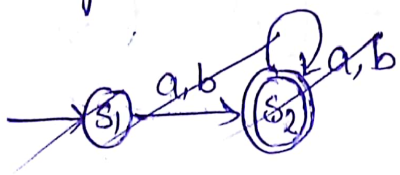
Scanned by CamScanner

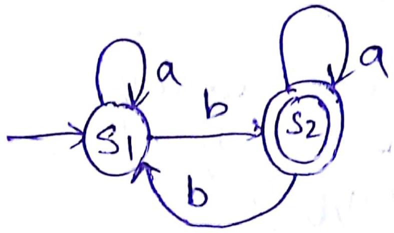
17. construct a Finite Automata where each string having odd no. of b's.

possible strings are:

(a, b) - b, ab, ba, bbb, aab, baa, aba,

State diagram:-

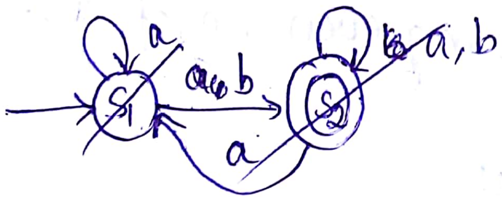


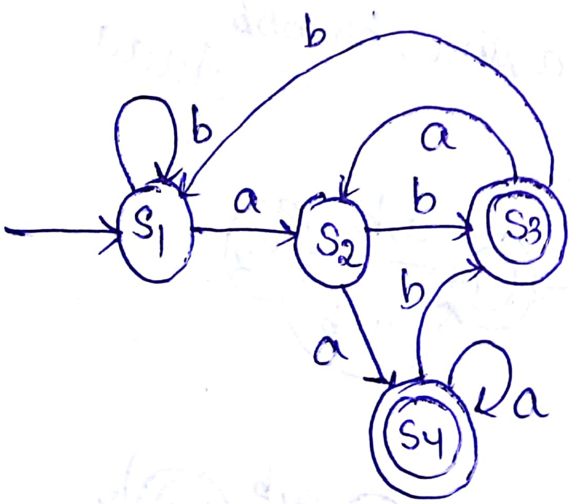


18. Construct a finite Automata on input symbol Σ(a,b) where each string having last but one symbol as 'a'.

Possible strings are:

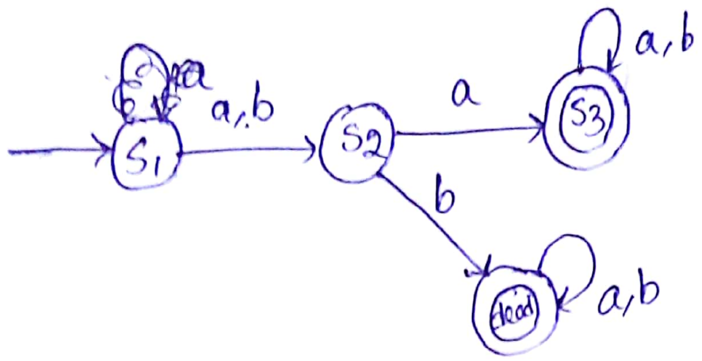
(a, b) - aa, ab, aab, bab, baa, aaaa, aaab, abab, baab, bbab, abaa, ...





(19.) Construct a finite Automata on input symbol $\Sigma(a,b)$ where each string having second symbol 'a'.

(a,b) - ba, aa, aab, bab, baa, aaaa, aaa, aaab,



(20.) construct a FA on $\Sigma(a,b)$ where each string having.

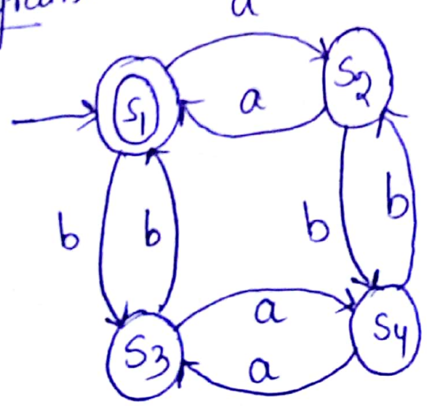
(1.) Even no. of a's (OR) Even no. of b's.

(2.) Even no. of a's and Even no. of b's.

(2.) The possible strings are:

(a,b) - ϵ, aa, bb, aaaa, aabb, abab, baab, baba, - - - -

State diagram:



(1.)

* Construct a Finite Automata on $\Sigma(0,1)$ where each binary string divisible by 2.

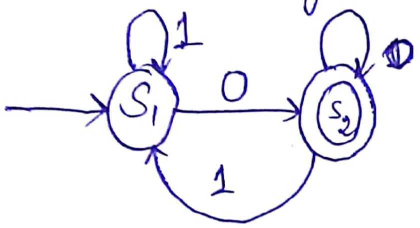
0000 - 0
0001 - 1
0010 - 2 ✓
0011 - 3
0100 - 4 ✓
0101 - 5
0110 - 6 ✓
1000 - 8 ✓

1. The possible strings are:

0010, 0100, 0110, 1000, 1010, - - - -

- - - - - -

2. The state diagram is:



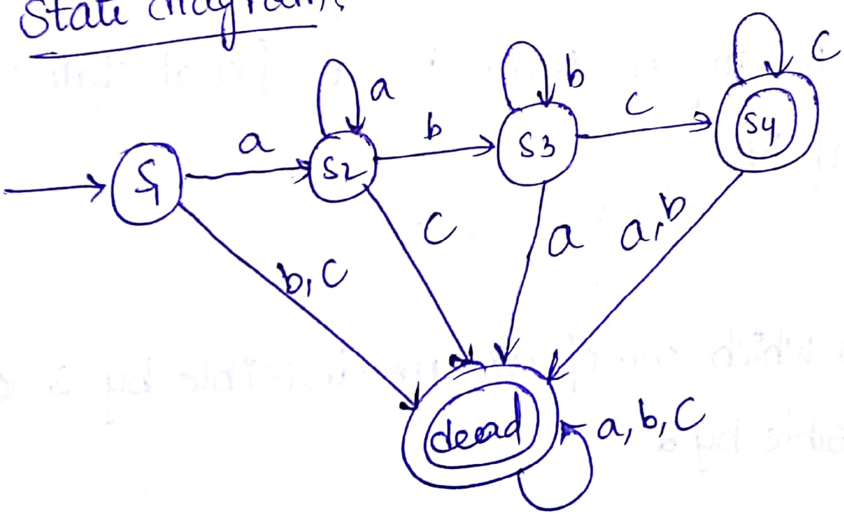
* Construct a FA on $\Sigma(a,b,c)$ where language

$l = a^l b^m c^n \mid l,m,n \geq 0$

The possible strings are:

abc, aabc, abbc, abcc, - - - - -

State diagram:

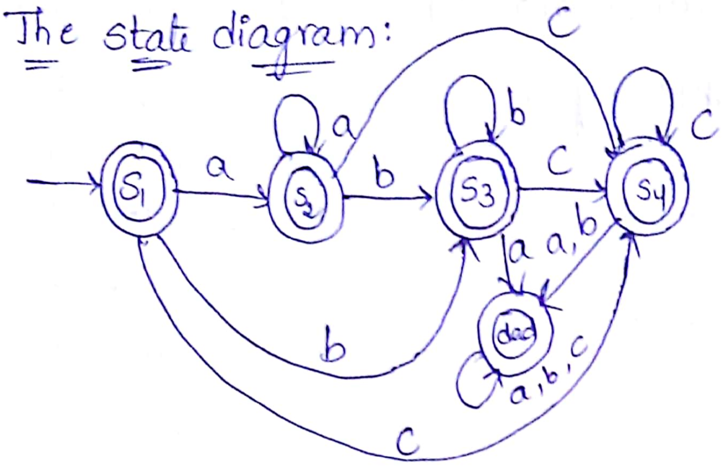


* $l = a^l b^m c^n ;\ l,m,n \geq 0$

The Possible Strings are:

(a,b) - ϵ, a, b, c, ab, ac, bc, aa, bb, cc, aaa, - - -

The state diagram:



Compound Finite Automata:

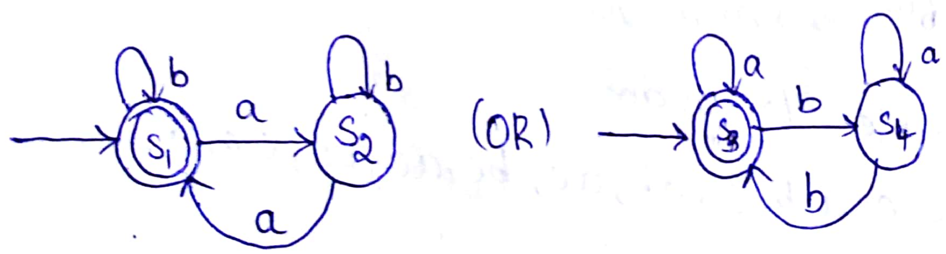
1. Union
2. Inter Section.
3. Set difference.
4. Complement.

(1.) Union:

If two Finite Automata's Combined with 'OR' then We have to make Final States of Compound Finite Automata if it is anyone state is a final state of anyone individual FA.

Eg:

Construct a FA in which no. of a's are divisible by 2 (or) no. of b's are divisible by 2.



no. of a's divisible by 2 no. of b's divisible by 2.

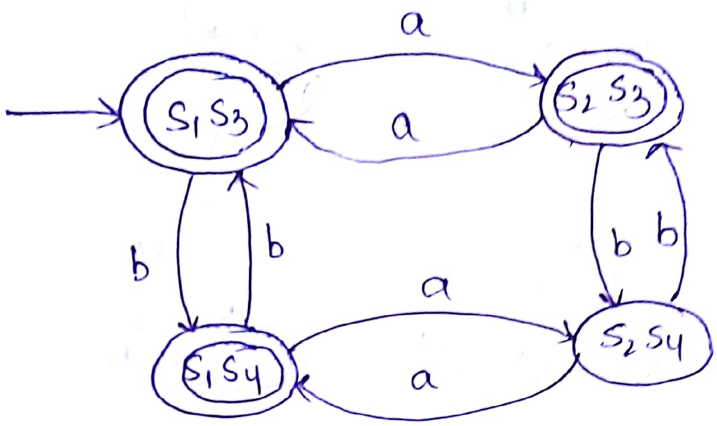
Starting state: Combined Starting state of both FA.

Final state: Final states in both FA is S_1, S_3.

So mark the S_1/S_3 states as final states.

$S_1 S_3$
$S_1 S_4$
$S_2 S_3$
$S_2 S_4$

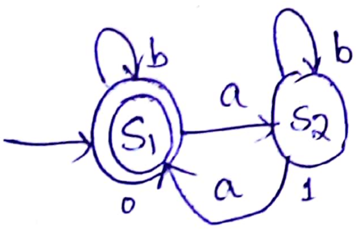
Compound FA:- no. of a's even (or) no. of b's even.

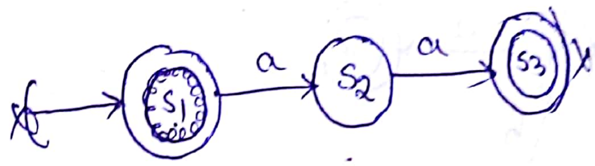


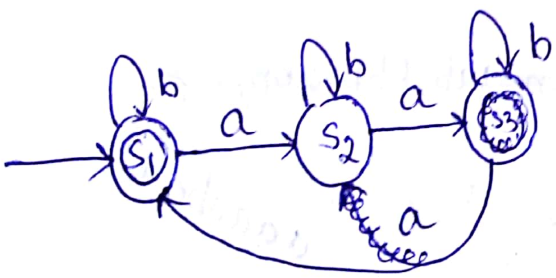
The Possible strings are:

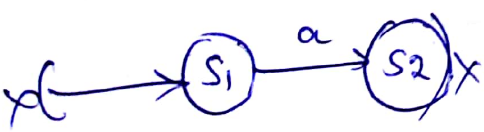
ϵ, a, b, aa, bb, aab, baa, aba, bba, ...

(2) Construct a Finite Automata where no. of a's are divisible by 2 (or) 3.

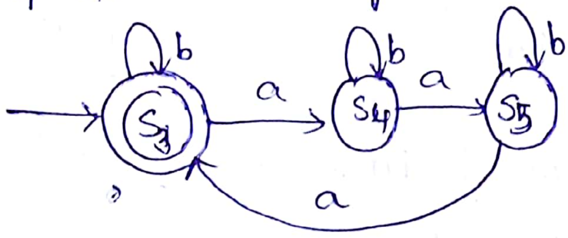






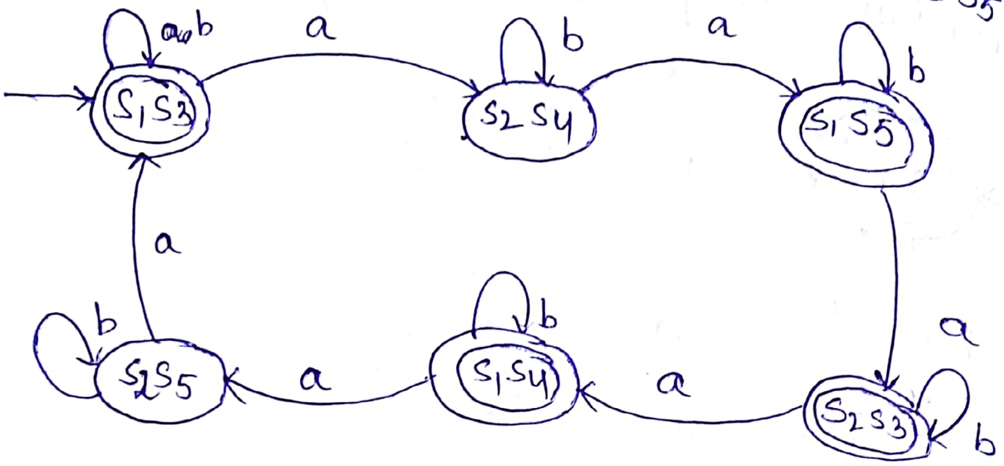


• no. of a's divisible by 3



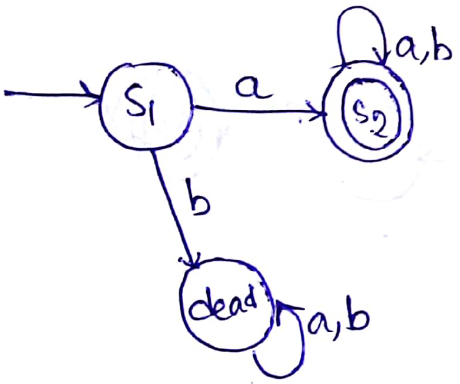
S_1S_3
S_1S_4
S_1S_5
S_2S_3
S_2S_4
S_2S_5

Compound Finite Automata:

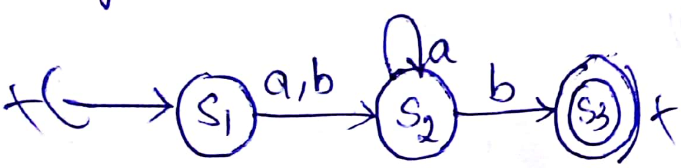


3. Construct a FA on input symbol $\Sigma(a,b)$ where ea[ch] String starts with a (or) ends with b.

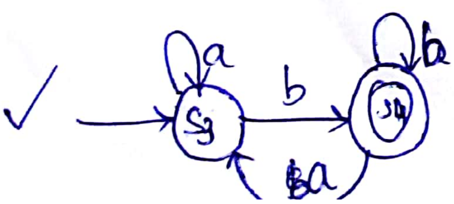
String start with 'a': a, ab, aa, aab, aba, abb, ...



String ends with 'b': b, ab, bb, aab, bbb, abb, ...

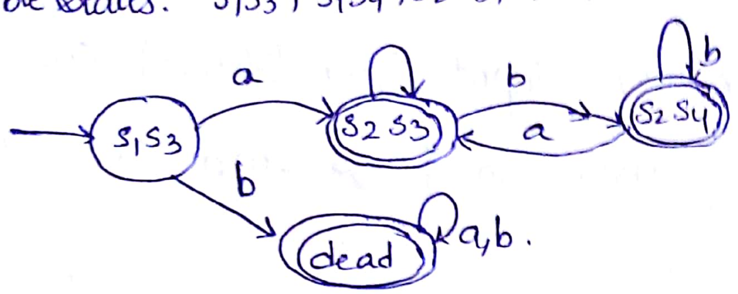


aaaabaab



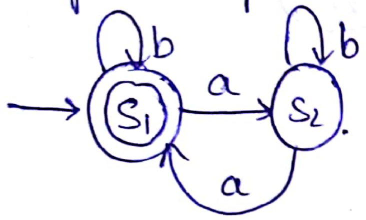
Possible States: S_1S_3, S_1S_4, S_2S_3, S_2S_4, dead.

(b)

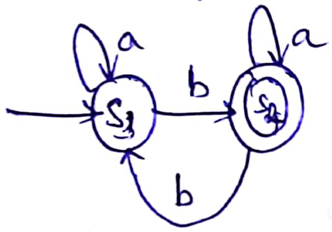


4. Construct a FA on input symbol a,b where each string having even no. of a's (or) odd no. of b's.

strings having even no. of a's :

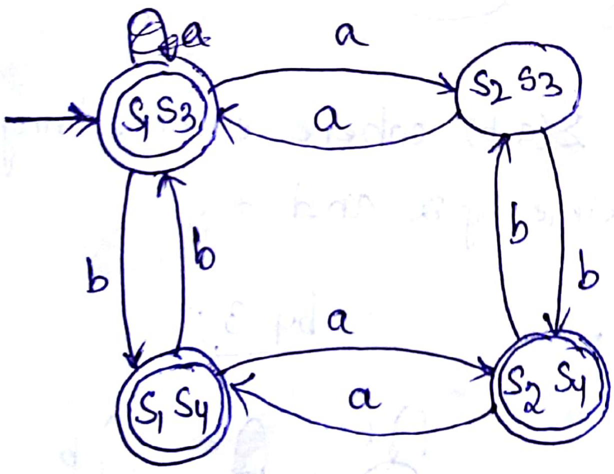


strings having odd no. of b's : b, ab, ba



Possible States are : S_1S_3, S_1S_4, S_2S_3, S_2S_4.

S_1, S_4



(2.)

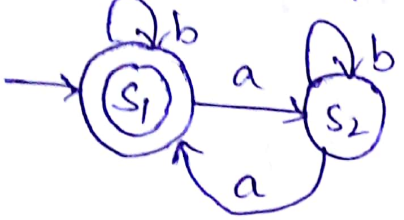
Intersection:

If any two FA are combined with 'AND' u have 2 make Final states of both FA's as Final states o combined FA.

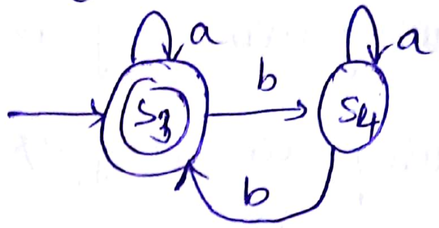
Q. Intersection:

1. Construct a FA on Input Symbol Σ(a, b) where each string having even no. of a's and even no. b's.

even no. of a's

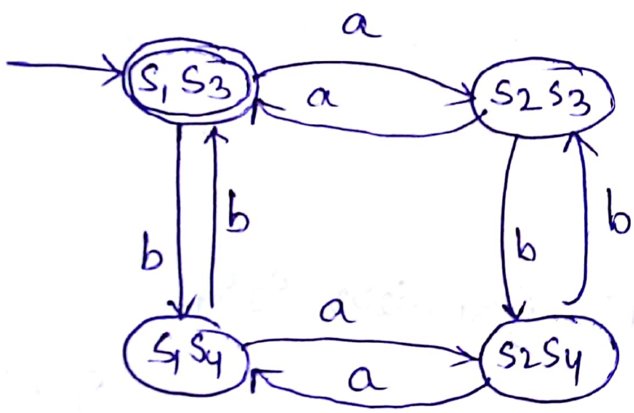


even no. of b's



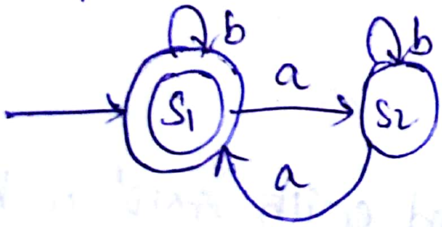
Compound FA:-

S_1S_3, S_1S_4, S_2S_3, S_2S_4

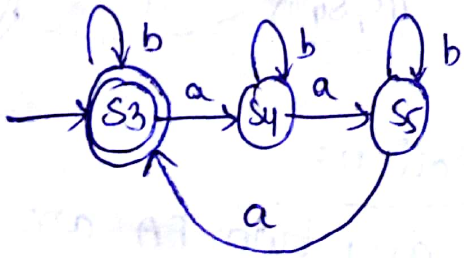


Q. Construct a FA on Σ(a, b) where each string having no. of a's divisible by 2 and 3.

no. of a's divisible by 2:

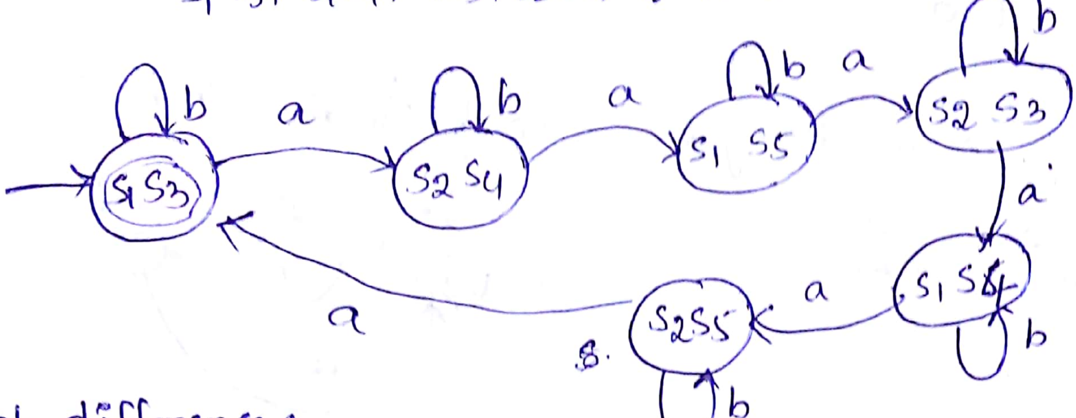


by 3:



Compound FA :-

$S_1 S_3$, $S_1 S_4$, $S_1 S_5$, $S_2 S_3$, $S_2 S_4$, $S_2 S_5$



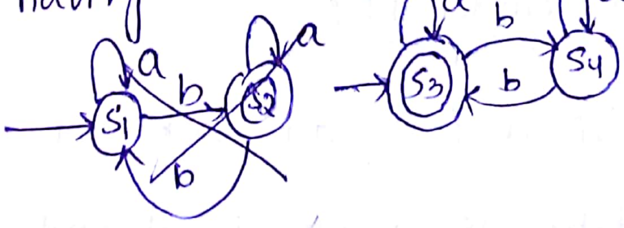
Set difference :

(3.) If two FA's combined with 'but not' we will make Final State as 1st FA but not having Final State of Second FA, as a Final state in Combined FA.

1. Construct a FA on Input symbol $\Sigma(a,b)$ where each string having even no. of a's but not even no. of b's.

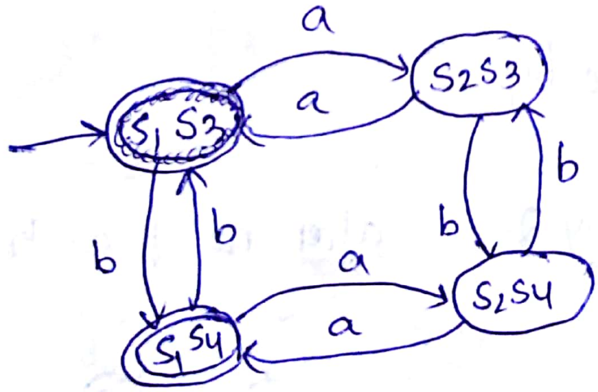
even no. of a's

but not having even no. of b's.



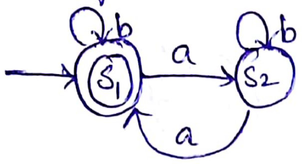
Compound FA :-

S_1 ✓
S_3 ×

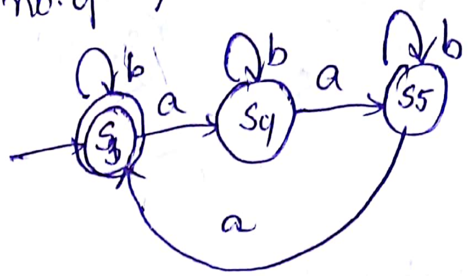


Q. Construct a FA on $\Sigma(a,b)$ no. of a's divisible by 2 but not 3.

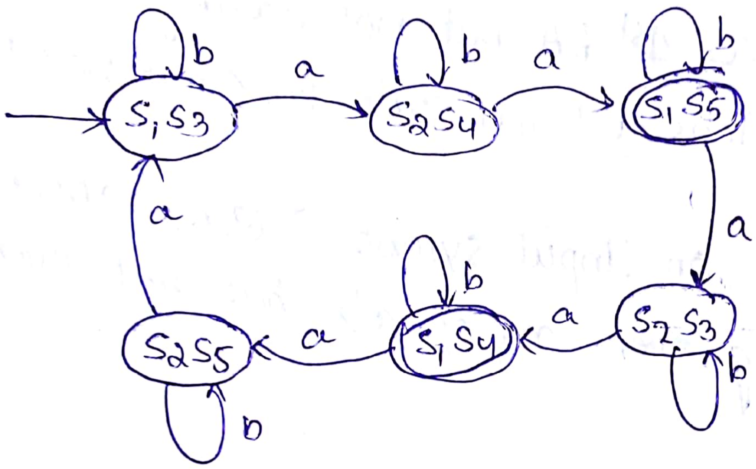
no. of a's divisible by 2



no. of a's divisible by 3



compound FA :-



S_1 ✓

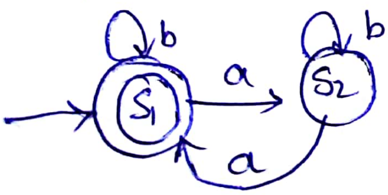
S_3 X

(4.) Complement :-

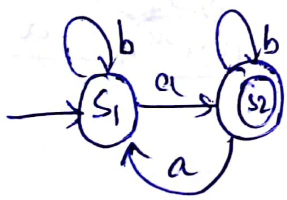
IF a FA starts with 'not' then make all the Final States as a non-Final and non-Final as Final States

1. Construct a FA on $\Sigma(a,b)$ where each string having no. of a's not divisible by 2.

no. of a's divisible by 2.



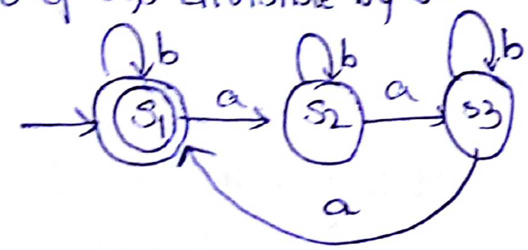
after no. of a's not divisible by 2.



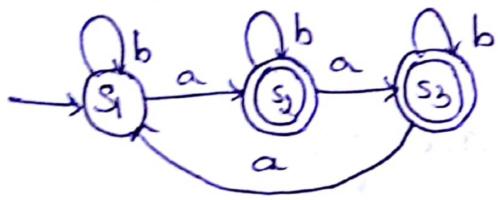
Scanned by CamScanner

2. Construct a FA on Σ(a,b) where each string having no of a's not divisible by 3.

no. of a's divisible by 3:

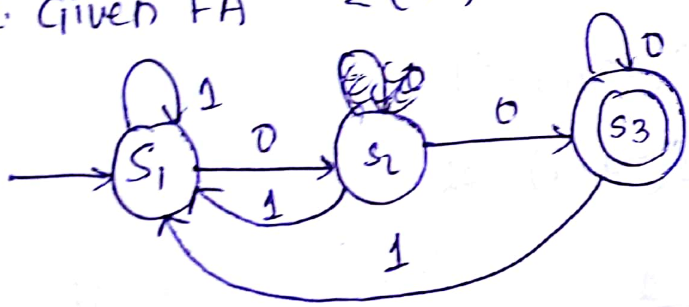


after no of a's not divisible by 3



Excersize:-

1. Given FA Σ (0,1). strings ends with '0'

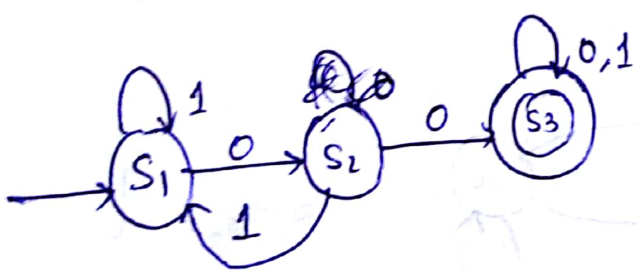


The possible strings are:

0, 00, 000, 1000, 1000, 0100, 1100, - - - - -

String with substring '

2.

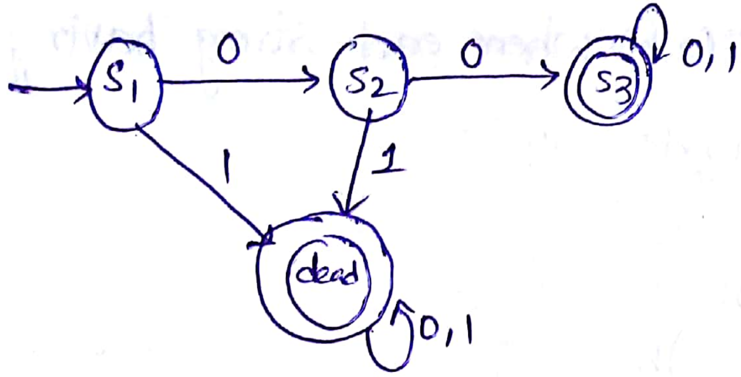


The Possible Strings are:

00, 100, 001100, 0100, 1001,

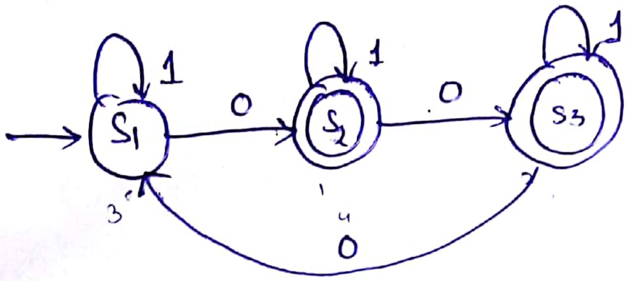
Scanned by CamScanner

3.



String starts with '00'.

4.



String with no. of '0's not divisible by 3.

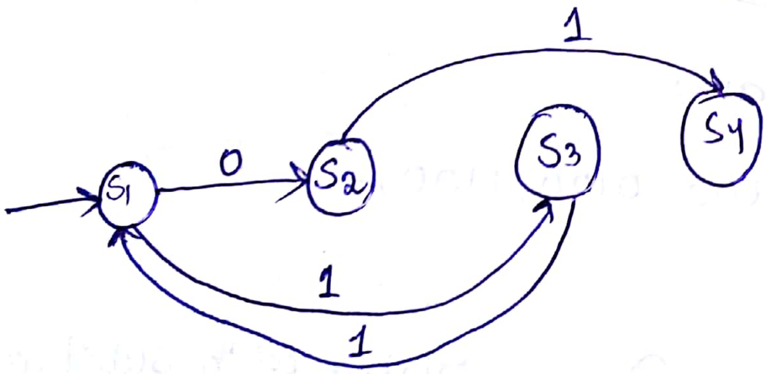
5. When set of Transistions are given:

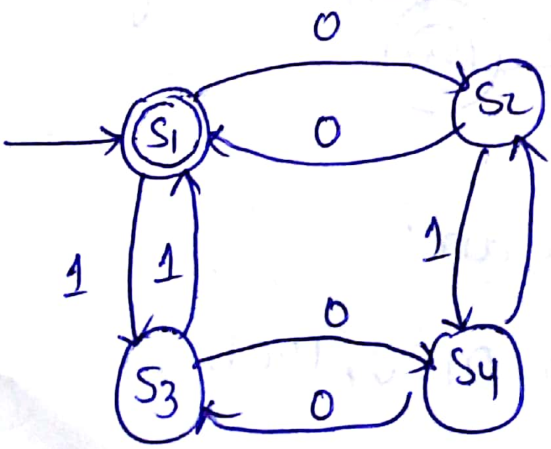
$\delta(s_1, 0) = s_2$ $\quad$ $\delta(s_1, 1) = s_3$

$\delta(s_2, 0) = s_1$ $\quad$ $\delta(s_2, 1) = s_4$

$\delta(s_3, 0) = s_4$ $\quad$ $\delta(s_3, 1) = s_1$

$\delta(s_4, 0) = s_3$ $\quad$ $\delta(s_4, 1) = s_2$

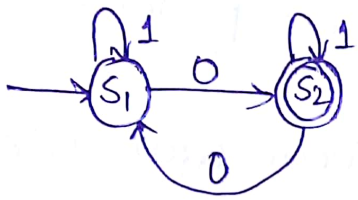




Scanned by CamScanner

6. Construct a FA on following language

$\mathcal{L} = 0, 000, 00000, \ldots$ on $\Sigma(0,1)$



Classification of Finite Automata:

Finite Automata

- FA without Output
  - DFA (design)
  - NFA (plan)
- FA with Output.
  - Meely Machine
  - Moore Machine

Non-Deterministic Finite Automata:

NFA is a 5 Tuple $(Q, \Sigma, \delta, S, F)$

Q - set of states.

Σ - set of input symbols.

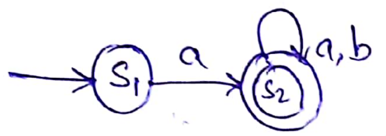
δ - Transistion from $Q \times \Sigma \rightarrow 2^Q$

where 2^Q is power set of 'Q'.

S - starting state.

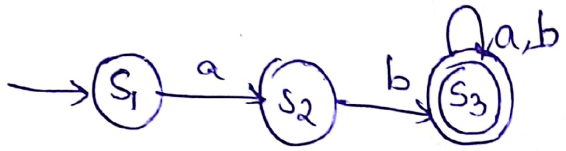
F - Final state.

1. Construct a NFA on $\Sigma(a,b)$ where each string starts with 'a'.



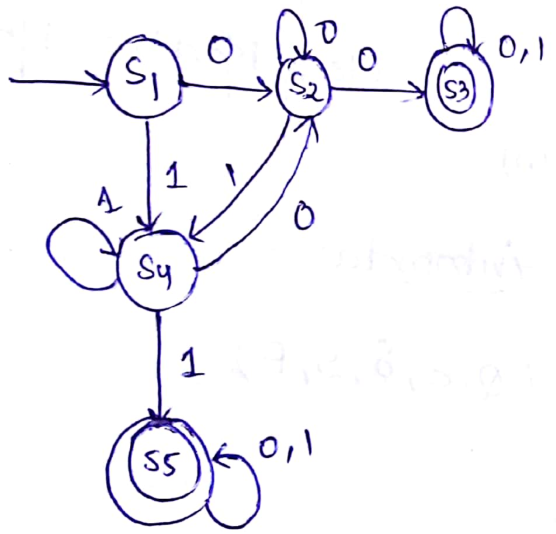
Drawing FA for possible strings avoiding not possible strings.

2. construct a NFA on $\Sigma(a,b)$ where each string starts with ab.

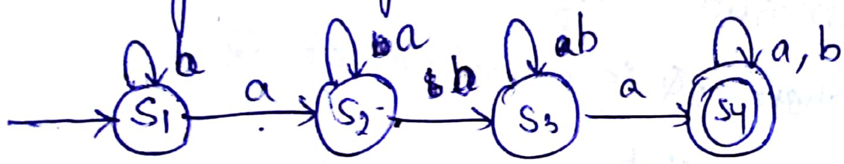


3. Construct a NFA on $\Sigma(0,1)$ where each string having consecutive '0's (or) 1's.

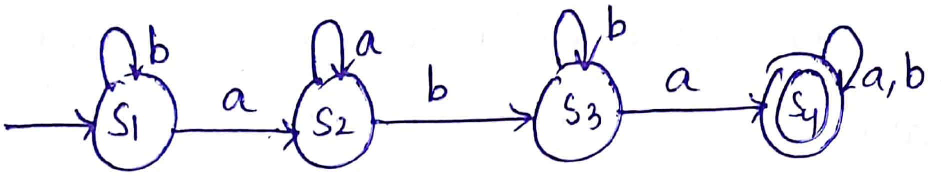
00, 11,



4. construct a NFA on input symbol $\Sigma(a,b)$ where each string having aba as a substring.



aababba

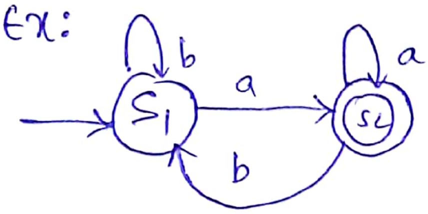


baabbaba

DFA

$Q \times \Sigma \rightarrow Q$

Ex:



$S_1 \times a \longrightarrow S_2$

(3) All the possibilities but not non-possibilities

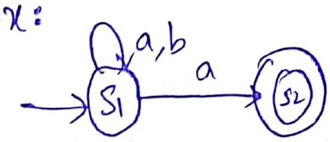
(1.) Transistion function

$Q \times \Sigma \longrightarrow Q$

NFA

$Q \times \Sigma \rightarrow 2^Q$

Ex:



$2^2 = 4$

$S_1 \times a \longrightarrow \phi,$
$S_1,$
$S_2,$
S_1, S_2

$2^{\text{no. of states}} = 2^2 = 4$

(3) Here the NFA accepts all the possibilities only.

(1.) Transistion function

$Q \times \Sigma \rightarrow 2^Q$

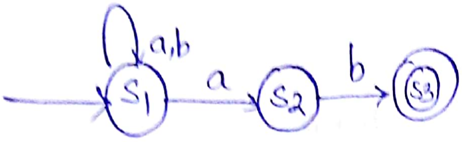
$2^Q \rightarrow$ power set of set of states.

(2) Note: The Power of DFA and NFA are both are same.

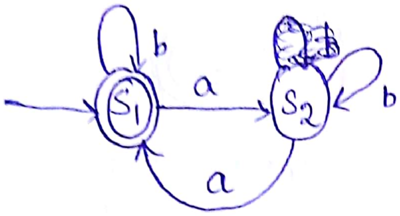
- By default all DFA's are NFA's.
- For each NFA We can construct equivalent DFA.

1. Construct a NFA on $\Sigma(a,b)$ where each string ends with ab.

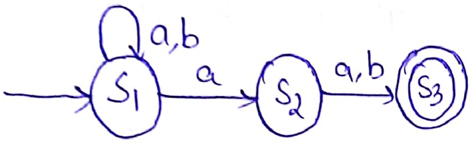
bab, ab, baab,



2. construct a NFA on $\Sigma(a,b)$ which accepts even no. of



3. construct a NFA on $\Sigma(a,b)$ where each string having last but one Symbol as a



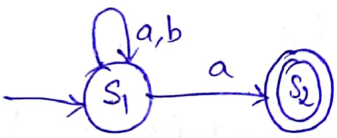
NFA to DFA Construction:-

construct a equivalent DFA for the following NFA.

(1.) where each string ends with 'a'.

Final state - S_2

NFA:

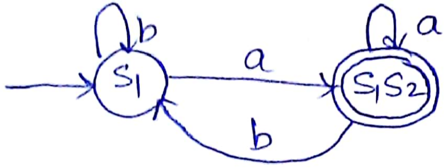


	a	b
S_1	$\{S_1,S_2\}$	S_1
S_2	$\{\ \}$	$\{\ \}$

Equivalent DFA:-

	a	b
S_1	$[S_1 S_2]$	S_1
$[S_1, S_2]$	$[S_1 S_2]$	S_1

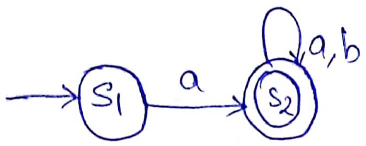
So, here consider, FS as S_2.



This is the Equivalent DFA.

2. Construct a Equivalent DFA for the following NFA. where each string starts with 'a'.

NFA:-

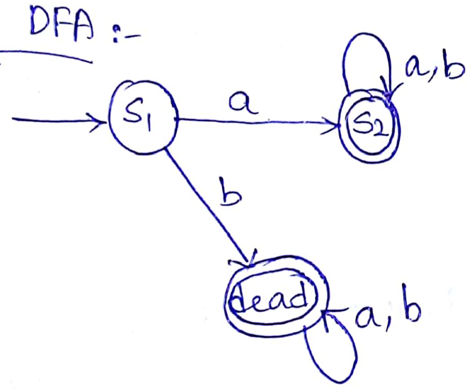


	a	b
S_1	S_2	{ }
S_2	S_2	S_2

Equivalent DFA

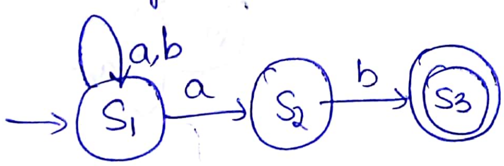
	a	b
S_1	S_2	dead
S_2	S_2	S_2
dead	dead	dead.

Required DFA :-



3. Construct an Equivalent DFA for the following NFA. where each string ends with 'ab'

~~NA~~ NFA:

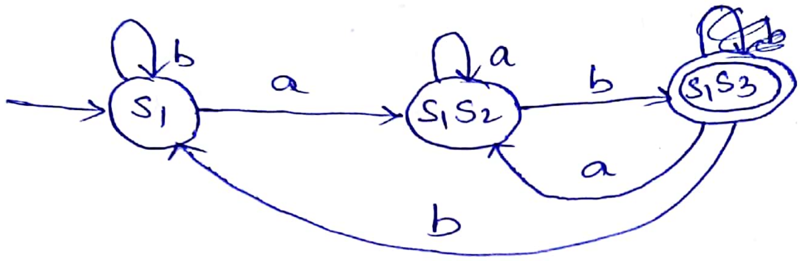


	a	b
S_1	$[S_1S_2]$	S_1
S_2	{ }	S_3
S_3	{ }	{ }

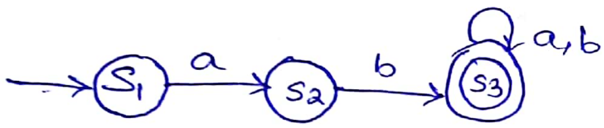
1. (Equivalent DFA :-

	a	b
S_1	$[S_1, S_2]$	S_1
$[S_1, S_2]$	$[S_1, S_2]$	$[S_1, S_3]$
$[S_1, S_3]$	$[S_1, S_2]$	~~$[S_1, S_3]$~~ S_1

2. (Required DFA :



3. 4; Construct a Equivalent DFA for the following NFA.

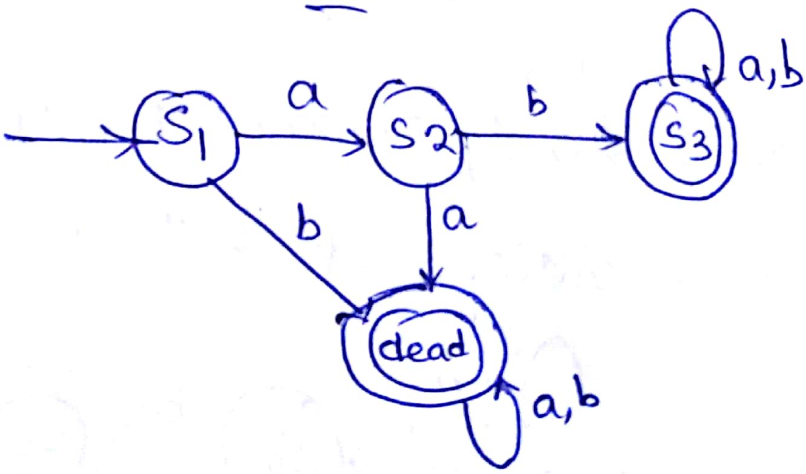


	a	b
S_1	S_2	{ } → dead.
S_2	{ }	S_3
S_3	S_3	S_3

For DFA :

	a	b
S_1	S_2	dead
S_2	dead	S_3
S_3	S_3	S_3

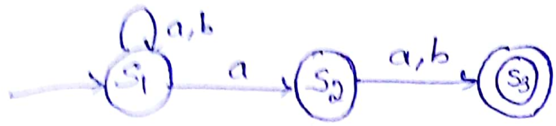
Required DFA :-



Scanned by CamScanner

5. Construct DFA for the following NFA.

String with last but one as 'a'.

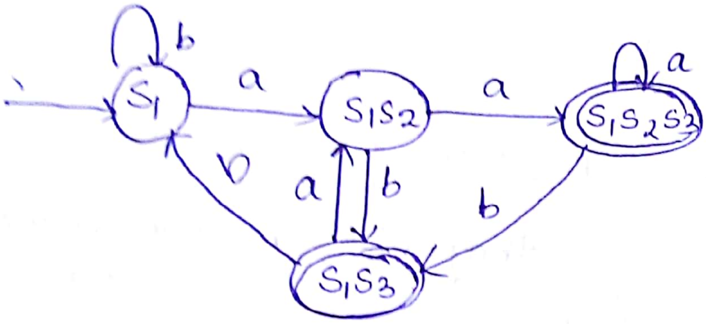


	a	b
S_1	$[S_1S_2]$	S_1
S_2	S_3	S_3
S_3	{ }	{ }

For DFA

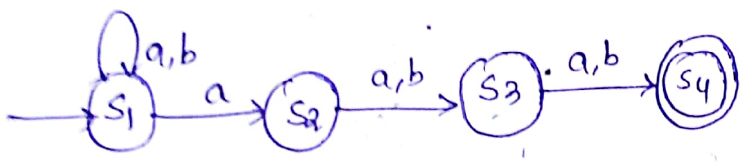
	a	b
S_1	$[S_1S_2]$	S_1
S_1S_2	$S_1S_2S_3$	$\{S_1S_3\}$
$S_1S_2S_3$	$S_1S_2S_3$	S_1S_3
S_1S_3	S_1S_2	S_1

Required DFA:-



6. Construct a Equivalent DFA for the following NFA.

String with left side 'a' 3rd.



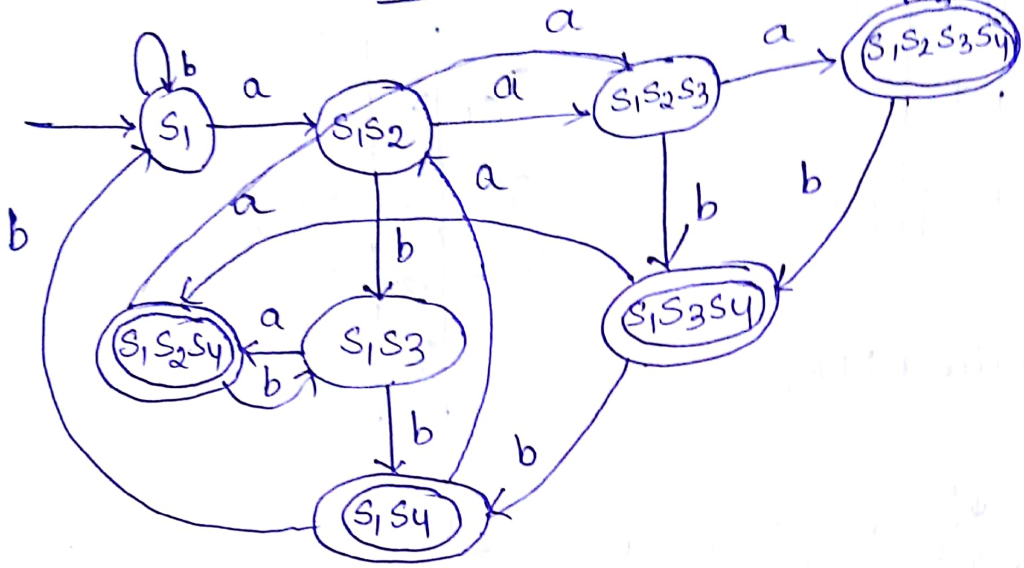
	a	b
S_1	$[S_1S_2]$	S_1
S_2	S_3	S_3
S_3	S_4	S_4
S_4	{ }	{ }

For DFA

	a	b
S_1	$\{S_1S_2\}$	S_1
$\{S_1S_2\}$	$S_1S_2S_3$	S_1S_3
$\{S_1S_2S_3\}$	$S_1S_2S_3S_4$	$S_1S_3S_4$
$\{S_1S_2S_3S_4\}$	$S_1S_2S_3S_4$	$S_1S_3S_4$
$\{S_1S_3\}$	$S_1S_2S_4$	S_1S_4
$\{S_1S_3S_4\}$	$S_1S_2S_4$	S_1S_4

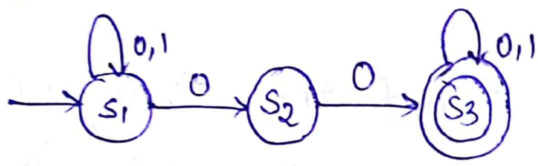
	a	b
{$S_1S_2S_4$}	$S_1S_2S_3$	S_1S_3
S_1S_4	S_1S_2	S_1

Required DFA



7. Construct Equivalent DFA for the following NFA.

Given string with two consecutive zero's

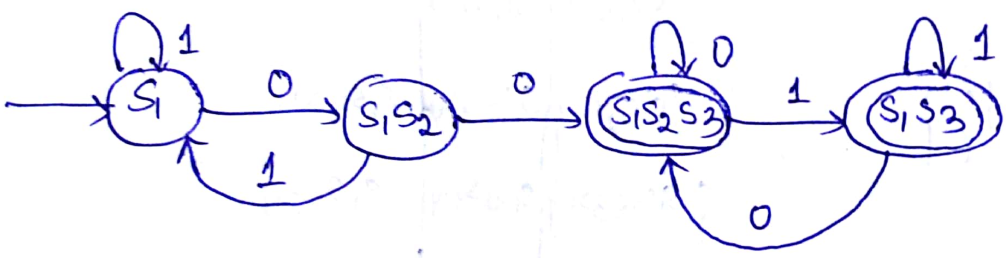


	0	1
S_1	[S_1S_2]	S_1
S_2	S_3	{ }
S_3	S_3	S_3

DFA

	0	1
S_1	[S_1S_2]	S_1
[S_1S_2]	$S_1S_2S_3$	S_1
$S_1S_2S_3$	$S_1S_2S_3$	S_1S_3
S_1S_3	$S_1S_2S_3$	S_1S_3

Required DFA :-



ϵ- NFA / Epsilon - NFA :

ϵ- NFA is a 5-Tuple $(Q, \Sigma, \delta, S, F)$

$Q \rightarrow$ set of states.

$\Sigma \rightarrow$ set of alphabets.

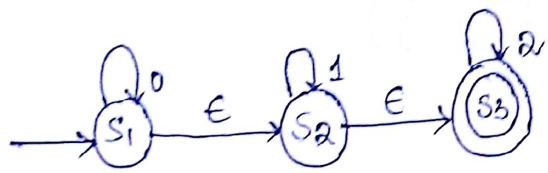
$\delta \rightarrow$ transition function on set of alphabets

$$Q \times (\Sigma \cup \epsilon) \rightarrow 2^Q$$

$S \rightarrow$ starting state.

$F \rightarrow$ Final State

1. Construct a equivalent NFA for the following ϵ-NFA



ϵ-closer(S_1) = (with out reading any alphabet from S_1 what states u can reach).

ϵ - Closer(S_1) = $\{S_1, S_2, S_3\}$

$\delta \rightarrow \epsilon$-NFA

$\delta' \rightarrow$ NFA

ϵ - Closer(S_2) = $\{S_2, S_3\}$

ϵ - closer(S_3) = $\{S_3\}$

$\delta'(S_1, 0) = \epsilon\text{-closer}\left(\delta(\epsilon\text{-closure}(S_1)), 0\right)$

$\delta'(S_1, 0) = \epsilon\text{-Closer}\left(\delta(S_1, S_2, S_3), 0\right)$

$= \epsilon\text{-closer}(S_1, \phi, \phi)$

$= \epsilon\text{-Closer}(S_1)$

$= \{S_1, S_2, S_3\}$

$$\delta'(s_1, 1) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_1)), 1)$$

$$= \epsilon\text{-closure}(\delta(s_1, s_2, s_3), 1)$$

$$= \epsilon\text{-closure}(\{\}, s_2, \{\})$$

$$= \epsilon\text{-closure}(s_2)$$

$$= \{s_2, s_3\}$$

$$\delta'(s_2, 0) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_2)), 0)$$

$$= \epsilon\text{-closure}(\delta(s_2, s_3), 0)$$

$$= \epsilon\text{-closure}(\{\}, \{\})$$

$$= \phi$$

$$\delta'(s_2, 1) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_2)), 1)$$

$$= \epsilon\text{-closure}(\delta(s_2, s_3), 1)$$

$$= \epsilon\text{-closure}(s_2, \{\})$$

$$= \{s_2, s_3\}$$

$$\delta'(s_3, 0) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_3)), 0)$$

$$= \epsilon\text{-closure}(\delta(s_3), 0)$$

$$= \epsilon\text{-closure}(\{\})$$

$$= \phi$$

$$\delta'(s_3, 1) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_3)), 1)$$

$$= \epsilon\text{-closure}(\delta(s_3), 1))$$

$$= \phi$$

$$\delta'(s_1, 2) = \epsilon\text{-closure}[\delta(\epsilon\text{-closure}(s_1)), 2)$$
$$= \epsilon\text{-closure}(\delta(s_1, s_2, s_3), 2)$$
$$= \epsilon\text{-closure}(\{\}, \{\}, s_3)$$
$$= \{s_3\}$$

$$\delta'(s_2, 2) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_2)), 2)$$
$$= \epsilon\text{-closure}(\delta(s_2, s_3), 2)$$
$$= \{s_3\}$$

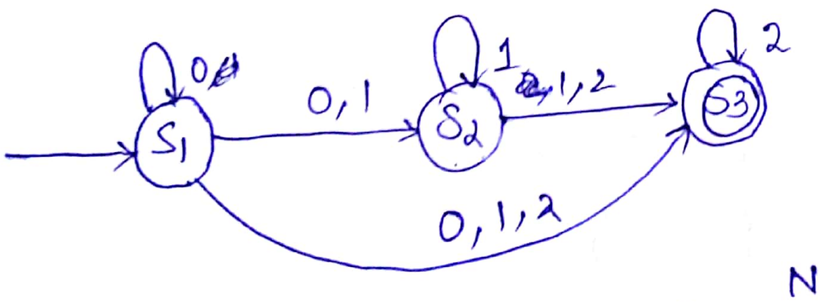
$$\delta'(s_3, 2) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_3)), 2)$$
$$= \epsilon\text{-closure}(\delta(s_3), 2)$$
$$= \{s_3\}$$

$\delta'(s_1, 0) = \{s_1, s_2, s_3\}$ $\quad$ $\delta'(s_2, 0) = \{\phi\}$ $\quad$ $\delta'(s_3, 0) = \{\phi\}$

$\delta'(s_1, 1) = \{s_2, s_3\}$ $\quad$ $\delta'(s_2, 1) = \{s_2, s_3\}$ $\quad$ $\delta'(s_3, 1) = \{\phi\}$

$\delta'(s_1, 2) = \{s_3\}$ $\quad$ $\delta'(s_2, 2) = \{s_3\}$ $\quad$ $\delta'(s_3, 2) = s_3$

Equivalent NFA:



NFA:

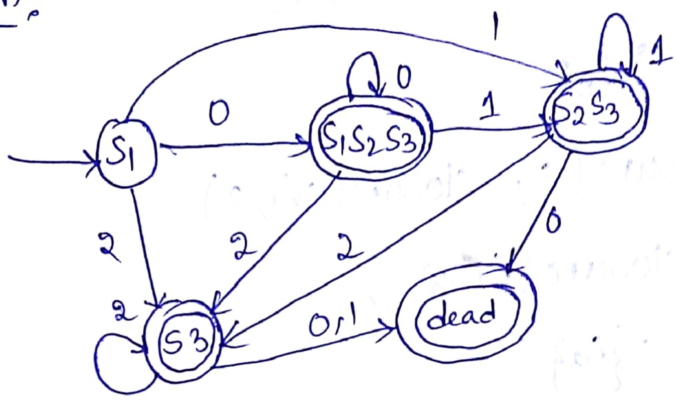
	0	1	2
s_1	$\{s_1 s_2 s_3\}$	$s_2 s_3$	s_3
s_2	ϕ	$s_2 s_3$	s_3
s_3	ϕ	ϕ	s_3

Scanned by CamScanner

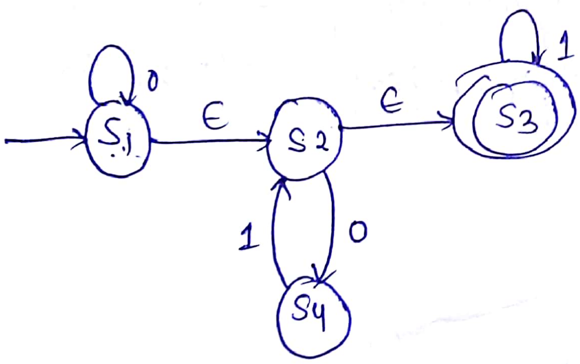
Equivalent DFA:

	0	1	2
S_1	$[S_1S_2S_3]$	$[S_2S_3]$	S_3
$[S_1S_2S_3]$	$[S_1S_2S_3]$	$[S_2S_3]$	S_3
$[S_2S_3]$	dead	$[S_2S_3]$	S_3
S_3	dead	dead	S_3

DFA:



2. Construct an Equivalent NFA for the following ϵ-NFA.



ϵ-closure$(S_1) = \{S_1, S_2, S_3\}$

ϵ-closure$(S_2) = \{S_2, S_3\}$

ϵ-closure$(S_3) = \{S_3\}$

ϵ-closure$(S_4) = \{S_4\}$

$$\delta'(s_1,0) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}\{s_1\}),0)$$

$$= \epsilon\text{-closure}(\delta(s_1, s_2, s_3),0)$$

$$= \epsilon\text{-Closure}(s_1, s_4, \{\})$$

$$= \{s_1, s_2, s_3, s_4\}$$

$$\delta'(s_1,1) = \epsilon\text{-Closure}(\delta(\epsilon\text{-closure}(s_1)),1)$$

$$= \epsilon\text{-Closure}(\delta(s_1, s_2, s_3),1)$$

$$= \epsilon\text{-Closure}(\{\},\{\},s_3)$$

$$= \{s_3\}$$

$$\delta'(s_2,0) = \epsilon\text{-closure}(\delta(\epsilon\text{-Closure}(s_2)),0)$$

$$= \epsilon\text{-Closure}(\delta(s_2, s_3),0)$$

$$= \epsilon\text{-Closure}(s_4,\{\})$$

$$= \{s_4\}$$

$$\delta'(s_2,1) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_2)),1)$$

$$= \epsilon\text{-Closure}(\delta(s_2, s_3),1)$$

$$= \epsilon\text{-Closure}(\{\},s_3)$$

$$= \{s_3\}$$

$$\delta'(s_3,0) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_3)),0)$$

$$= \epsilon\text{-closure}(\delta(s_3),0)$$

$$= \{\} = \phi$$

$\delta'(s_3,1) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_3)),1)$

$= \epsilon\text{-closure}(\delta(s_3),1)$

$= \epsilon\text{-closure}(s_3)$

$= \{s_3\}$

$\delta'(s_4,0) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_4)),0)$

$= \epsilon\text{-closure}(\delta(s_4),0)$

$= \{\ \}$

$\delta'(s_4,1) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(s_4)),1)$

$= \epsilon\text{-closure}(\delta(s_4),1)$

$= \epsilon\text{-closure}(s_2)$

$= \{s_2, s_3\}$

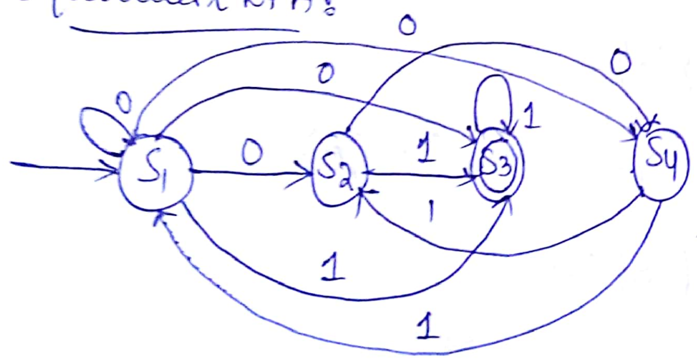
$\delta'(s_1,0) = \{s_1 s_2 s_3 s_4\}$ $\qquad$ $\delta'(s_3,0) = \phi$

$\delta'(s_1,1) = \{s_3\}$ $\qquad$ $\delta'(s_3,1) = \{s_3\}$

$\delta'(s_2,0) = \{s_4\}$ $\qquad$ $\delta'(s_4,0) = \{\ \}$

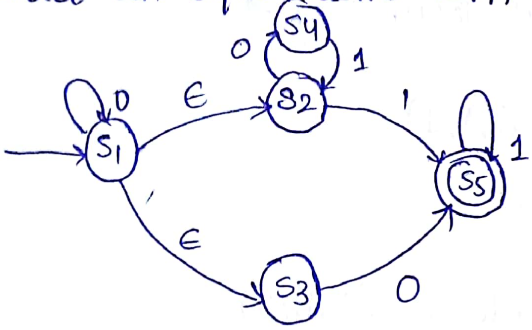
$\delta'(s_2,1) = \{s_3\}$ $\qquad$ $\delta'(s_4,1) = \{s_2, s_3\}$

Equivalent NFA:



	0	1
s_1	$\{s_1 s_2 s_3 s_4\}$	$\{s_3\}$
s_2	s_4	s_3
s_3	$\{\ \}$	s_3
s_4	$\{\ \}$	$[s_1 s_2]$

3. Construct an Equivalent NFA for the following ϵ-NFA.



$\epsilon\text{-closure}(S_1) = \{S_1, S_2, S_3\}$

$\epsilon\text{-closure}(S_2) = \{S_2\}$

$\epsilon\text{-closure}(S_3) = \{S_3\}$

$\epsilon\text{-closure}(S_4) = \{S_4\}$

$\epsilon\text{-closure}(S_5) = \{S_5\}$

$$\delta'(S_1, 0) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(S_1)), 0)$$
$$= \epsilon\text{-closure}(\delta(S_1, S_2, S_3), 0)$$
$$= \epsilon\text{-closure}(S_1, S_4, \{S_5\})$$
$$= \{S_1, S_2, S_3, S_4, S_5\}$$

$$\delta'(S_1, 1) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(S_1)), 1)$$
$$= \epsilon\text{-closure}(\delta(S_1, S_2, S_3), 1)$$
$$= \epsilon\text{-closure}(\{\}, S_5, \{\})$$
$$= \{S_5\}$$

$$\delta'(S_2, 0) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(S_2)), 0)$$
$$= \epsilon\text{-closure}(\delta(S_2), 0)$$
$$= \epsilon\text{-closure}(S_4)$$
$$= \{S_4\}$$

$$\delta'(S_2,1) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(S_2)),1)$$
$$= \epsilon\text{-closure}(\delta(S_2),1)$$
$$= \{S_5\}$$
$$\delta'(S_3,0) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(S_3)),0)$$
$$= \epsilon\text{-closure}(\delta(S_3),0)$$
$$= \{S_5\}$$
$$\delta'(S_3,1) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(S_3)),1)$$
$$= \epsilon\text{-closure}(\delta(S_3),1)$$
$$= \{\} = \phi$$
$$\delta'(S_4,0) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(S_4)),0)$$
$$= \epsilon\text{-closure}(\delta(S_4),0)$$
$$= \{\}$$
$$\delta'(S_4,1) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(S_4)),1)$$
$$= \epsilon\text{-closure}(\delta(S_4),1)$$
$$= \{S_2\}$$
$$\delta'(S_5,0) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(S_5)),0)$$
$$= \epsilon\text{-closure}(\delta(S_5),0)$$
$$= \{\}$$
$$\delta'(S_5,1) = \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(S_5)),1)$$
$$= \epsilon\text{-closure}(\delta(S_5),1)$$
$$= \{S_5\}$$

$\delta'(S_1,0) = \{S_1, S_2, S_3, S_4, S_5\}$

$\delta'(S_1,1) = \{S_5\}$

$\delta'(S_2,0) = \{S_4\}$

$\delta'(S_2,1) = \{S_5\}$

$\delta'(S_3,0) = \{S_5\}$

$\delta'(S_3,1) = \{\phi\}$

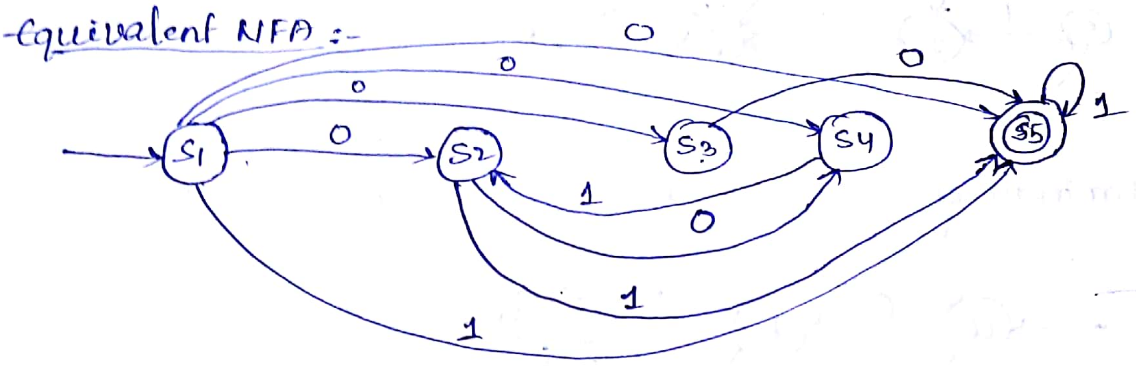
$\delta'(S_4,0) = \{\phi\}$

$\delta'(S_4,1) = \{S_2\}$

$\delta'(S_5,0) = \{\}$

$\delta'(S_5,1) = \{S_5\}$

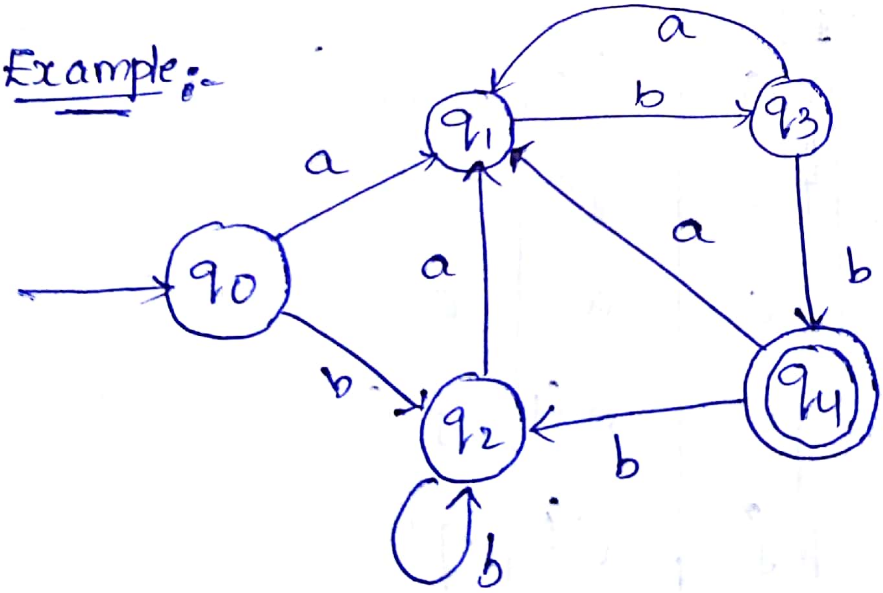
Equivalent NFA :-

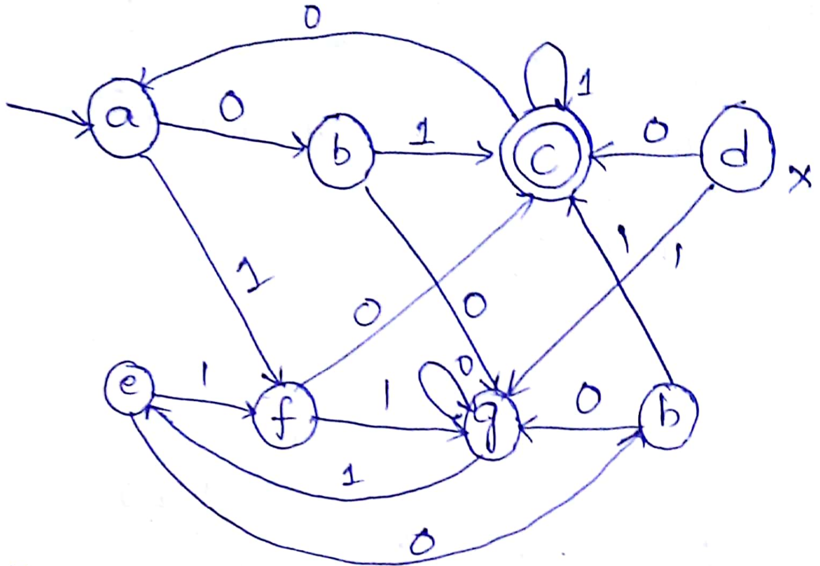


Minimisation of DFA:

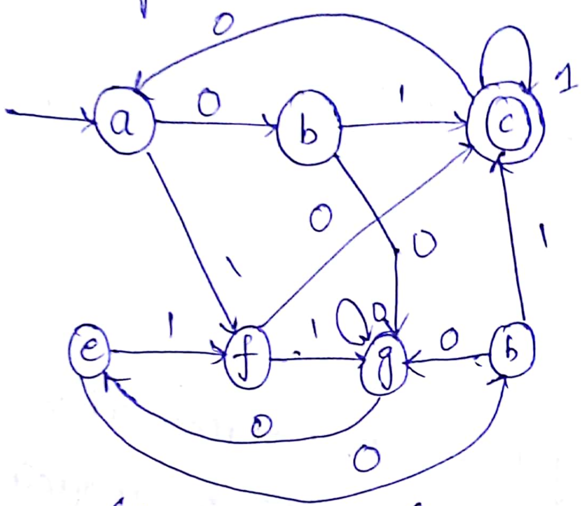
IF a DFA having more no. of states then we can construct Equivalent Minimal DFA by reducing no. of states

• Example:-





By eliminating unreachable states like 'd' in above,



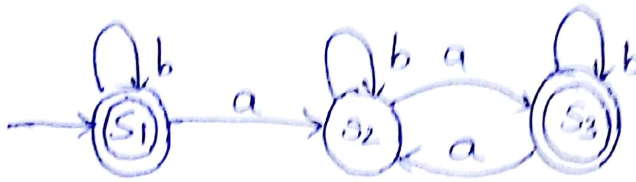
$\Pi_0 = \{\overbrace{a, b, e, g, h, f}^{C_1}\}$ $\overbrace{\{c\}}^{C_2}$

non Final   Final

$\Pi_1 = \{a, e, g\}$ $\{b\}$ $\{f\}$

	0	1
a	b C_1	f C_1
b	g C_1	c C_2
c	a	c
e	h C_1	f C_1
f	c C_2	g C_1
g	g	e
h	g	c

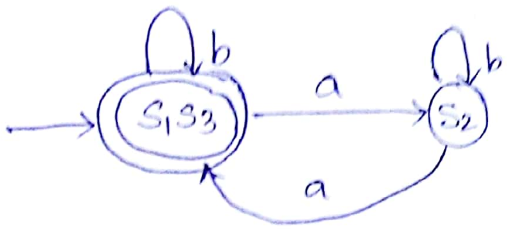
1.



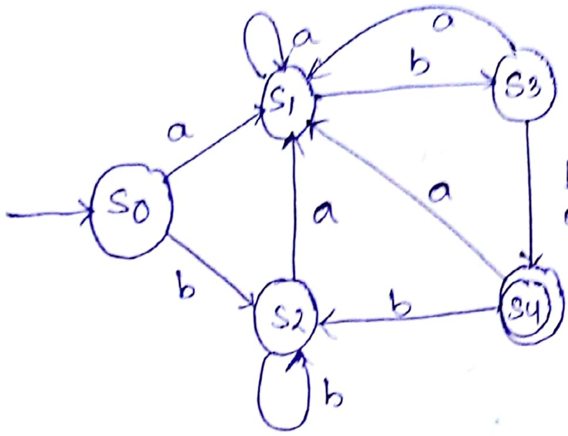
	a	b
s_1	s_2	s_1
s_2	s_3	s_2
s_3	s_2	s_3

$\Pi_0 = \{s_2\} \ \{s_1, s_3\}$

$\Pi_1 = \{s_2\} \ \{s_1, s_3\}$



2.



	a	b	
s_0	s_1	s_2	c_1
s_1	s_1	s_3	$c_1 \mid c_3$
s_2	s_1	s_2	c_1
s_3	s_1	s_4	c_2
s_4	s_1	s_2	

$\Pi_0 = \{\underbrace{s_0, s_1, s_2, s_3}_{c_1}\} \cdot \{\underbrace{s_4}_{c_2}\}$

$\Pi_1 = \{s_0, s_1, s_2\}_{c_1} \quad \{s_3\}_{c_2} \quad \{s_4\}_{c_3}$

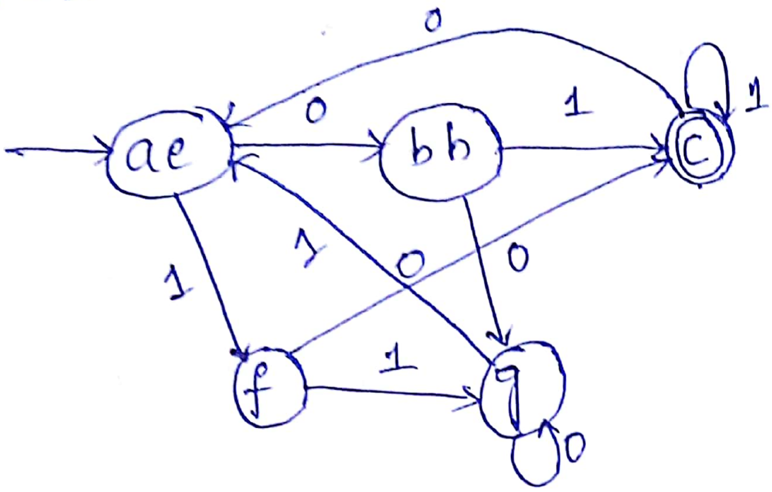
$\Pi_2 = \{s_0, s_2\}_{c_1} \quad \{s_1\}_{c_2} \quad \{s_3\}_{c_3} \quad \{s_4\}_{c_4}$

$\Pi_3 = \{s_0, s_2\} \ \{s_1\} \ \{s_3\} \ \{s_4\}$

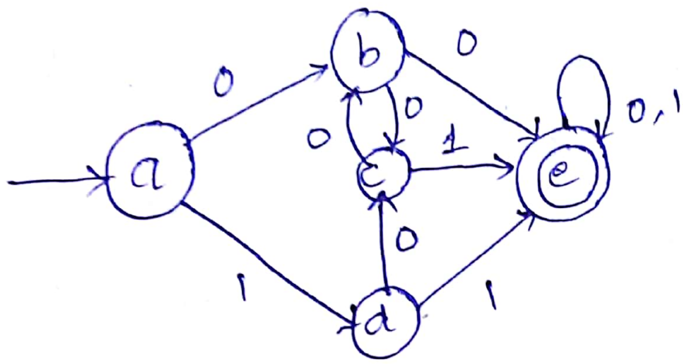
$\because \Pi_2 = \Pi_3$

Stop the Process.

Minimal Finite Automata:



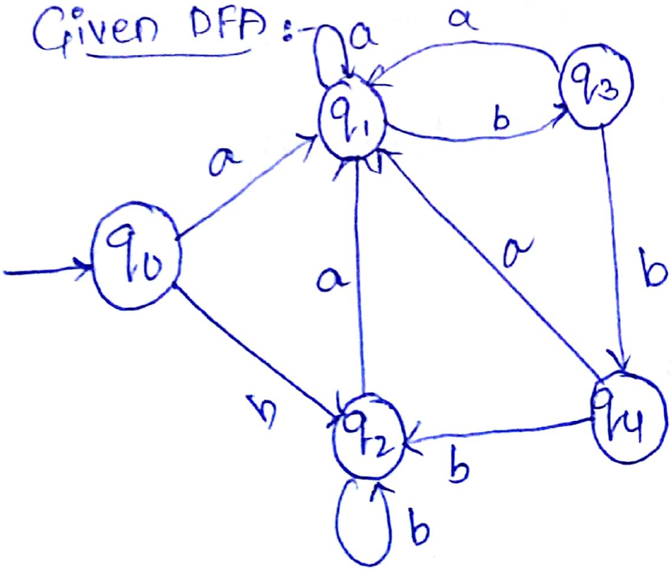
4.



My hill Nerode theorems: (Table Filling Method).

* Construct the Equivalent Minimal DFA for the Following DFA by using Myhill Nerode theorem.

Given DFA:-



	q_0	q_1	q_2	q_3
q_0				
q_1	✓			
q_2		✓		
q_3	✓	✓	✓	
q_4	✓	✓	✓	✓

$\delta(q_1,a)=q_1$ }
$\delta(q_0,a)=q_1$ } No need to

$\delta(q_1,b)=q_3$ }
$\delta(q_0,b)=q_2$ }

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

$\delta(q_2,b)=q_2$ }
$\delta(q_0,b)=q_2$ }

$\delta(q_1,a)=q_1$ }
$\delta(q_2,a)=q_1$ }

$\delta(q_1,b)=q_3$

$\delta(q_2,b)=q_2$

compare q_3, q_0

$\delta(q_3,a)=q_1$ }
$\delta(q_0,a)=q_1$ }

$\delta(q_3,b)=q_4$ } Mark the
$\delta(q_0,b)=q_2$ } cell

compare q_3, q_1

$\delta(q_3,a)=q_1$ }
$\delta(q_1,a)=q_1$ }

$\delta(q_3,b)=q_4$ } Mark
$\delta(q_1,b)=q_3$ } the cell

compare q_2, q_3

$\delta(q_3,a)=q_1$ }
$\delta(q_2,a)=q_1$ }

$\delta(q_3,b)=q_4$ } Mark
$\delta(q_2,b)=q_2$ } the cell

Compare q_1, q_0

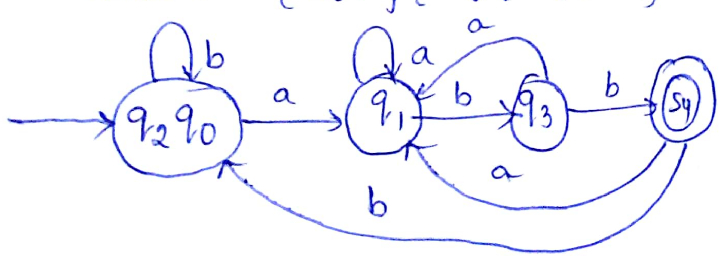
$\delta(q_1,a)=q_1$ }
$\delta(q_0,a)=q_1$ }

$\delta(q_1,b)=q_3$ }
$\delta(q_0,b)=q_3$ }

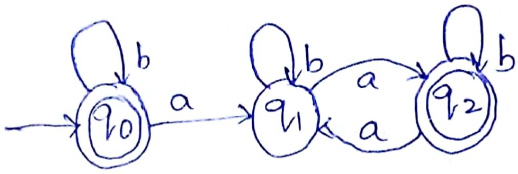
compare q_2, q_0

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

∴ The States are $\{q_2 q_0\}$ $\{q_1\}$ $\{q_3\}$ $\{q_4\}$



2.



compare q_2, q_0

$$\left.\begin{array}{l}\delta(q_0, a) = q_1 \\ \delta(q_2, a) = q_1\end{array}\right\}$$

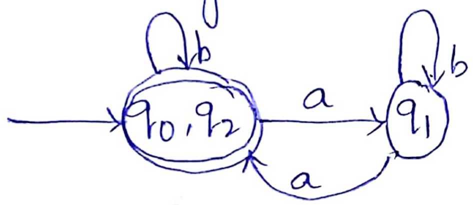
$\delta(q_0, b) = q_0$

$\delta(q_2, b) = q_2$

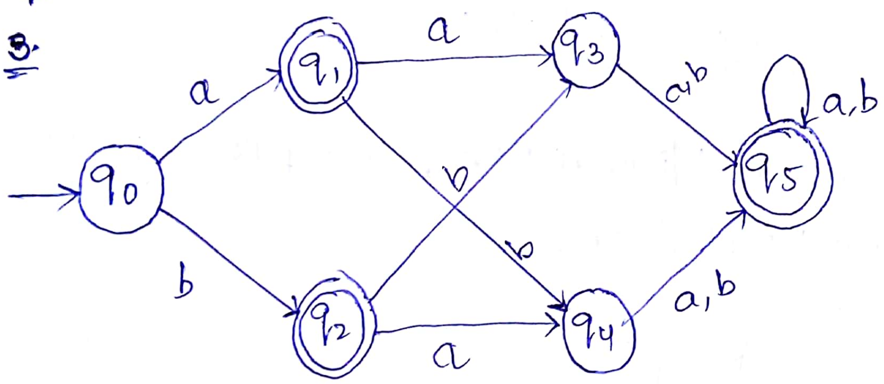
	q_0	q_1	q_2
q_0			
q_1	✓		
q_2		✓	

$\therefore \{q_1\} \{q_2, q_0\}$

The state diagram



3.



	q_0	q_1	q_2	q_3	q_4	q_5
q_0						
q_1	✓					
q_2	✓					
q_3	✓	✓	✓			
q_4	✓	✓	✓			
q_5	✓	✓	✓	✓	✓	

Scanned by CamScanner

Compare q_2, q_1

$\delta(q_1, a) = q_3$
$\delta(q_2, a) = q_4$ } No need

$\delta(q_1, b) = q_4$
$\delta(q_2, b) = q_3$ } No need

Compare q_3, q_0

$\delta(q_0, a) = q_1$
$\delta(q_3, a) = q_5$ } No mark

$\delta(q_0, b) = q_2$
$\delta(q_3, b) = q_5$ } No mark.

Compare q_5, q_2

$\delta(q_2, a) = q_4$
$\delta(q_5, a) = q_5$

$\delta(q_2, b) = q_3$
$\delta(q_5, b) = q_5$

Compare q_4, q_0

$\delta(q_0, a) = q_1$
$\delta(q_4, a) = q_5$ } No mark

$\delta(q_0, b) = q_2$
$\delta(q_4, b) = q_5$ } No mark

Compare q_4, q_3

$\delta(q_3, a) = q_5$
$\delta(q_4, a) = q_5$

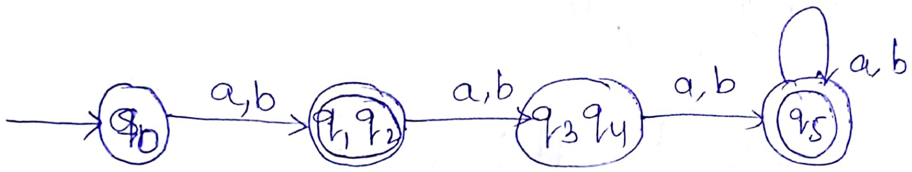
$\delta(q_3, b) = q_5$
$\delta(q_4, b) = q_5$

Compare q_5, q_1

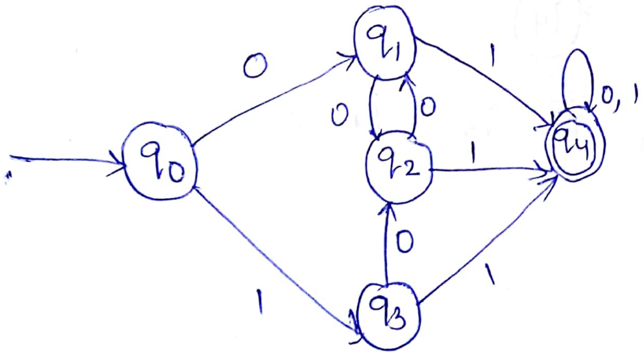
$\delta(q_1, a) = q_3$
$\delta(q_5, a) = q_5$

$\delta(q_5, b) = q_5$
$\delta(q_1, b) = q_4$

$\therefore \{q_0\}, \{q_1 q_2\}, \{q_4, q_3\}, \{q_5\}$



4. Construct Equivalent Minimal DFA for the following DFA by using Myhill Neard theorem.



Scanned by CamScanner

	q_0	q_1	q_2	q_3	q_4
q_0					
q_1	✓				
q_2	✓				
q_3	✓				
q_4	✓	✓	✓	✓	

$\delta(q_{10}$

$q_4,\ q_1q_3,\ q_2q_3,\ q_2q_1$

$q_4,\ q_1q_2q_3,\ q_0$

Compare q_0, q_1

$\delta(q_0, 0) = q_1$
$\delta(q_1, 0) = q_2$
$\delta(q_1, 1) = q_4$
$\delta(q_0, 1) = q_3$

Compare q_0, q_2

$\delta(q_0, 0) = q_1$
$\delta(q_2, 0) = q_1$
$\delta(q_2, 1) = q_4$
$\delta(q_0, 1) = q_3$

Compare q_1, q_2

$\delta(q_1, 0) = q_2$
$\delta(q_2, 0) = q_1$
$\delta(q_1, 1) = q_4$
$\delta(q_2, 1) = q_4$

Compare q_3, q_0

$\delta(q_0, 0) = q_1$
$\delta(q_3, 0) = q_4$
$\delta(q_0, 1) = q_3$
$\delta(q_3, 1) = q_4$

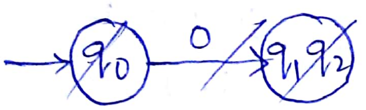
Compare q_1, q_3

$\delta(q_1, 0) = q_2$
$\delta(q_1, 1) = q_4$
$\delta(q_3, 0) = q_2$
$\delta(q_3, 1) = q_4$

Compare q_2, q_3

$\delta(q_2, 0) = q_1$
$\delta(q_3, 0) = q_2$
$\delta(q_2, 1) = q_4$
$\delta(q_3, 1) = q_4$

$\therefore \{q_0\}, \{q_1, q_2\}, \{q_2, q_3\}, \{q_3, q_1\}, \{q_4\}$

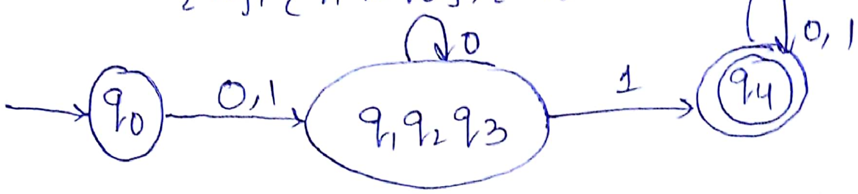


$q_1q_2q_3$

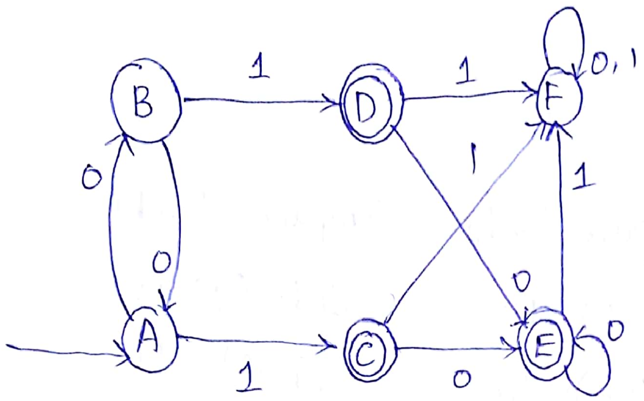
$\therefore \{q_0\}, \{q_1 q_2 q_3\} \{q_4\}$

The State diagram is:

$\{q_0\}, \{q_1 q_2 q_3\}, \{q_4\}$



5. Construct an Equivalent Minimal DFA for the following DFA.



	A	B	C	D	E	F
A						
B						
C	✓	✓				
D	✓	✓				
E	✓	✓		✓		
F	✓	✓	✓	✓	✓	

Compare A, B

$\left.\begin{array}{l}\delta(A,0) = B \\ \delta(B,0) = A\end{array}\right\}$

$\left.\begin{array}{l}\delta(A,1) = C \\ \delta(B,1) = D\end{array}\right\}$

Compare D, C

$\left.\begin{array}{l}\delta(D,0) = E \\ \delta(C,0) = E\end{array}\right\}$

$\left.\begin{array}{l}\delta(D,1) = F \\ \delta(C,1) = F\end{array}\right\}$

Compare E, C

$\left.\begin{array}{l}\delta(E,0) = E \\ \delta(C,0) = E\end{array}\right\}$

$\left.\begin{array}{l}\delta(E,1) = F \\ \delta(C,1) = F\end{array}\right\}$

Compare E, D

$\left.\begin{array}{l}\delta(E,0) = E \\ \delta(D,0) = E\end{array}\right\}$

$\delta(E,1) = F$

$\delta(D,1) = E$

Compare A, F

$\delta(A,0) = B$
$\delta(F,0) = F$

$\delta(A,1) = C$
$\delta(F,1) = F$ ✓

Compare B, F

$\delta(B,0) = A$
$\delta(F,0) = F$

$\delta(B,1) = D$
$\delta(F,1) = F$

FINITE AUTOMATA WITH OUTPUT:

Moore Machine: [purpose: Counting]

Output of Moore Machine depends on present state. Moore Machine defined as 6-tuple

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

$Q \rightarrow$ Set of states

$\Sigma \rightarrow$ set of input alphabets.

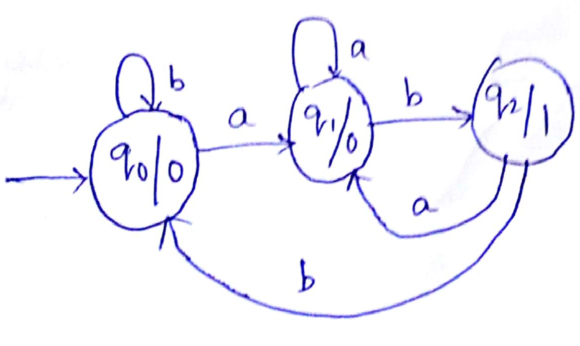
$\Delta \rightarrow$ set of Output alphabets.

$\delta \rightarrow$ set of Transistions having $Q \times \Sigma \rightarrow Q$

$\lambda \rightarrow$ set of Transistions Mapping $Q \rightarrow \Delta$

$q_0 \rightarrow$ Starting state.

(1) Construct a Moore Machine on $\Sigma(a,b)$ and which produces 1 for each Sequence of ab Otherwise produces 0.



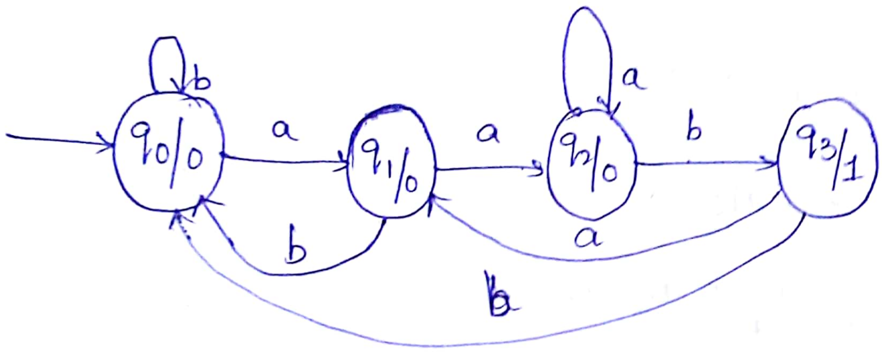
	Σ(a,b) a	b	Δ(0,1)
q_0	q_1	q_0	0
q_1	q_1	q_2	0
q_2	q_1	q_0	1

$Q - (q_0, q_1, q_2)$

$\Sigma = (a, b)$

q_0 - starting state.

2. Construct a MM which produces 1 for sequence of aab and produces '0' Otherwise.



	Σ(a,b) a	b	Δ(0,1)
q_0	q_1	q_0	0
q_1	q_2	q_0	0
q_2	q_2	q_3	0
q_3	q_1	q_0	1

$Q = (q_0, q_1, q_2, q_3)$

$\Sigma = (a, b)$

q_0 - Starting state.

Scanned by CamScanner

Meely Machine :-

Meely Machine Produces Output based on present state and present input.

Meely Machine is defined as 6-typle.

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

$Q \rightarrow$ Set of States.

$\Sigma \rightarrow$ set of input alphabets.

$\Delta \rightarrow$ set of Ouput alphabets.

$\delta \rightarrow$ transistion Mapping $Q \times \Sigma \rightarrow Q$

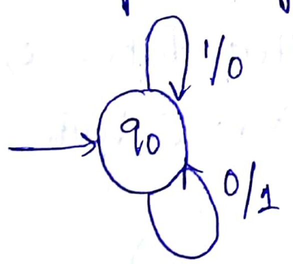
$\lambda \rightarrow$ transistion Mapping $Q \times \Sigma \rightarrow \Delta$

q_0 - starting state.

1. Construct a Meely Machine which produces Complement of a given binary string

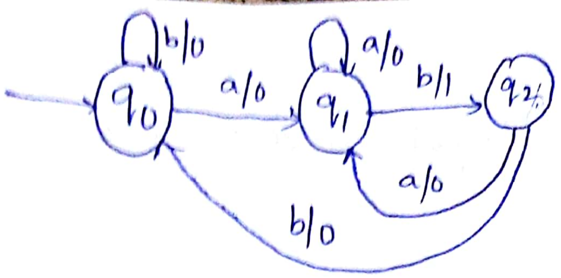
1100

0011



	$\Sigma(0,1)$		$\Delta(0,1)$	
	0	1	0	1
q_0	q_0	q_0	1	0

2. Construct Meely Machine which Counts Occurrences of ab.

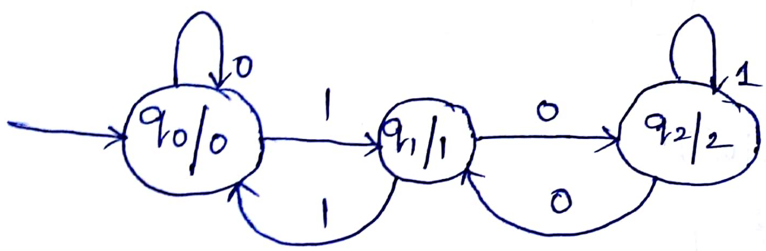


	Σ(0,1) 0	1	Δ(0,1) a	b
q_0	q_1	q_0	0	0
q_1	q_1	q_2	0	1
q_2	q_1	q_0	0	0

101
011

3. Construct a Meely Machine which produces 2's complement for a given input binary string.

4. Construct a Moore Machine that takes 0,1 as input and produces residuo Module 3 as Output.



	Σ(~~a,b~~ 0,1) 0	1	Δ(0,1) Output
q_0	q_0	q_1	0
q_1	q_2	q_0	1
q_2	q_1	q_2	2

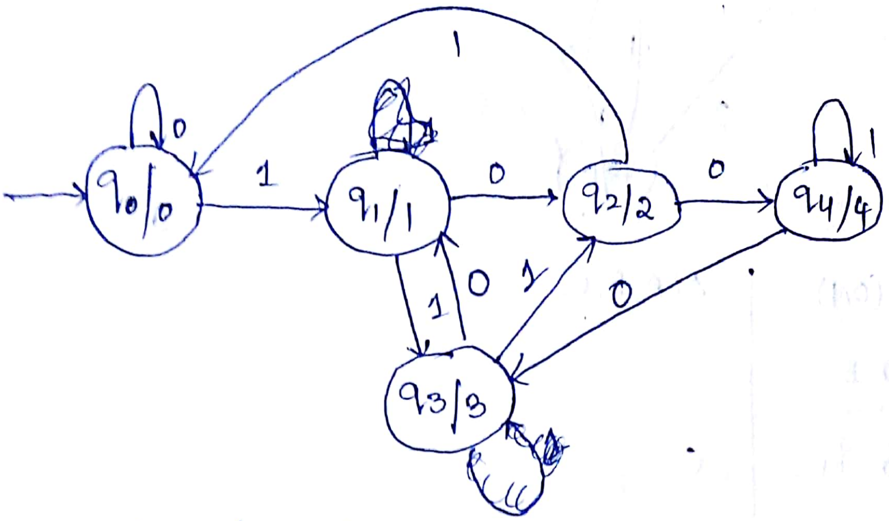
$Q = \{q_0, q_1, q_2\}$

$\Sigma = (0,1)$

$\Delta = (0,1,2)$

$S - q_0$

5. Construct Moore Machine that takes 0,1 as input and produces ~~residue~~ Module 5 as Output.



0001
0010
0011
$1 \div 5 = 1$
$2 \div 5 = 2$
0011
$3 \div 5 = 3$
0100
$4 \div 5 = 4$
0000
$5 \div 5 = 0$
0110
$6 \div 5 = 1$
$7 \div 5 = 2$
0111.
$8 \div 5 = 3$
1000
$9 \div 5 = 4$
1001

	$\Sigma(0,1)$ 0	1	Output
q_0	q_0	q_1	0
q_1	q_2	q_3	1
q_2	q_3	q_0	2
q_3	q_1	q_2	3
q_4	q_3	q_4	4

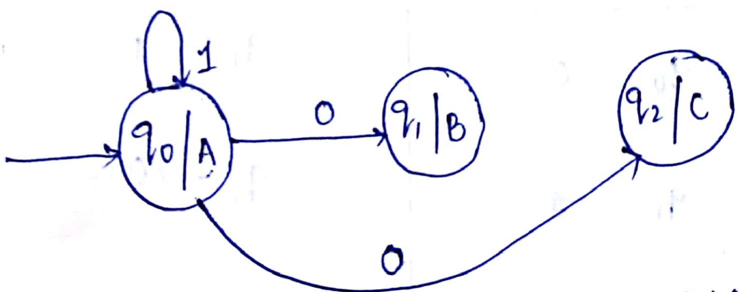
$Q = \{q_0, q_1, q_2, q_3, q_4\}$

$\Sigma = (0,1)$

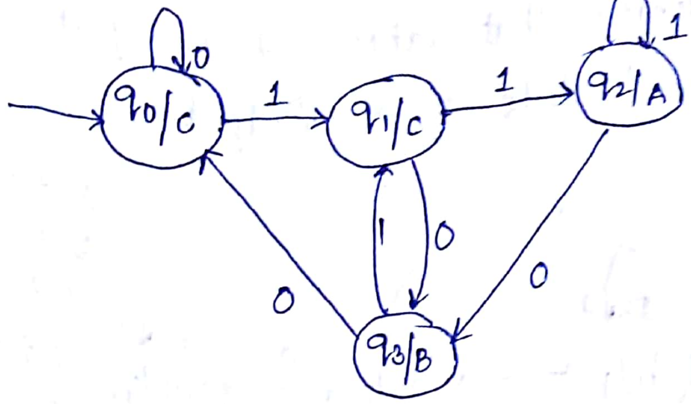
$\Delta = (0,1,2,3,4)$

$S - q_0$

6. Construct a Moore Machine which produces 'A' as Output for each Occurrence of 11 and produces Output 'B' for each Occurrence of 10. Otherwise produces 'C' for each Occurrence.

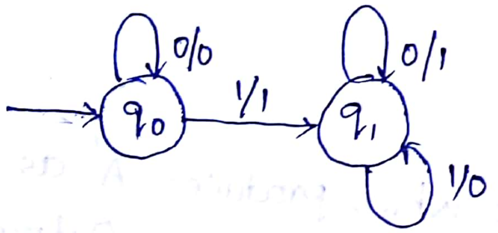


010110010
↓↓↓↓↓↓↓↓↓
C C B C A B C C B



	$\Sigma(0,1)$ 0	1	$\Delta(A,B,C)$
q_0	q_0	q_1	C
q_1	q_3	q_2	C
q_2	q_3	q_2	A
q_3	q_0	q_1	B

7. Construct Meely Machine which takes 0,1 as input and produces 2's complement as Output.

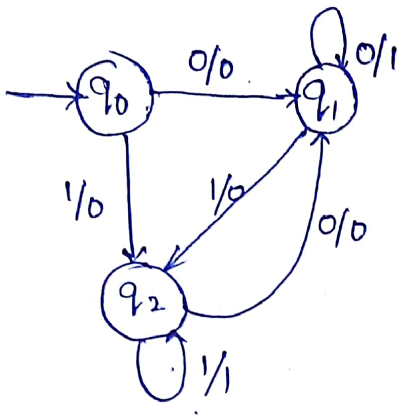


101101 ←

Present state	Next state i=0	Output	i=1	Output
q_0	q_0	0	q_1	1
q_1	q_1	1	q_1	0

Scanned by CamScanner

8. Construct Meely Machine that prints '1' if the last two symbols are same. Otherwise produces '0'.

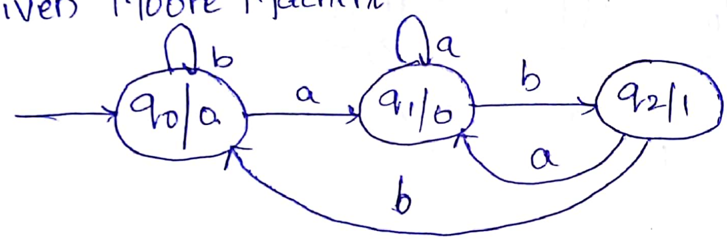


- Moore Machine to Meely Machine Conversion:

→ The power of Meely Machine & Moore Machine are same.

→ We can convert from one machine to another Machine.

Given Moore Machine

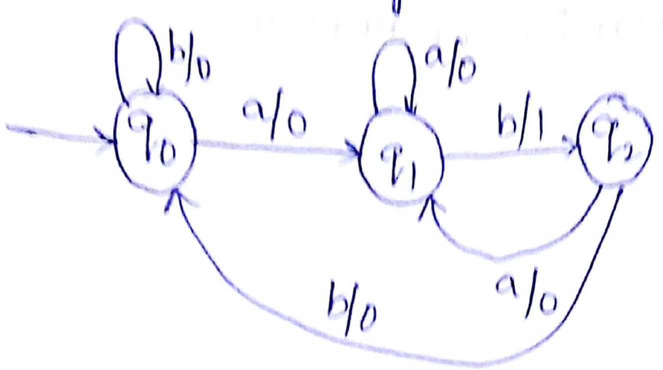


Present	Next i/p = a	o/p = b	Output
q_0	q_1	q_0	0
q_1	q_1	q_2	0
q_2	q_1	q_0	1

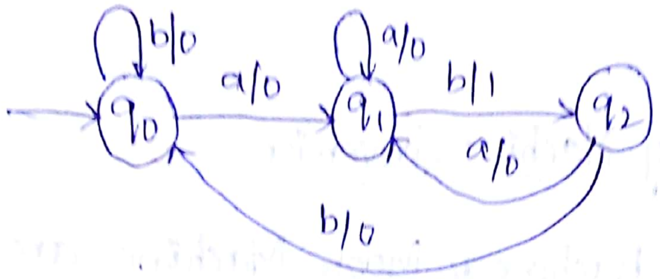
Present State	Next state i/p = a	o/p =	i/p = b	o/p
q_0	q_1	0	q_0	0
q_1	q_1	0	q_2	1
q_2	q_1	0	q_0	0

Scanned by CamScanner

The Equivalent Meely Machine :

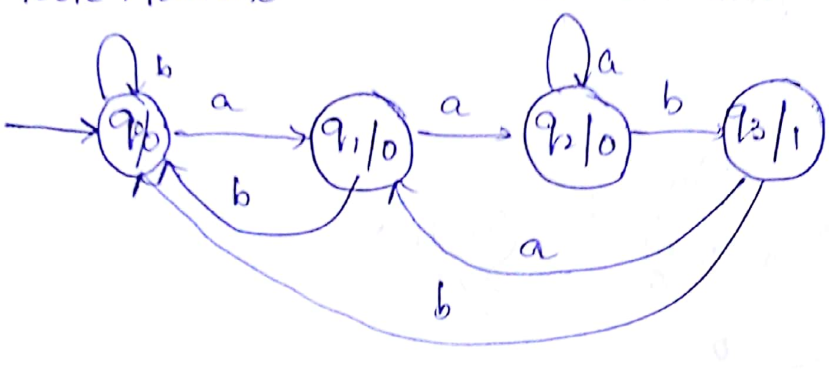


-Another Method directly from Moore Machine to Meely Machine



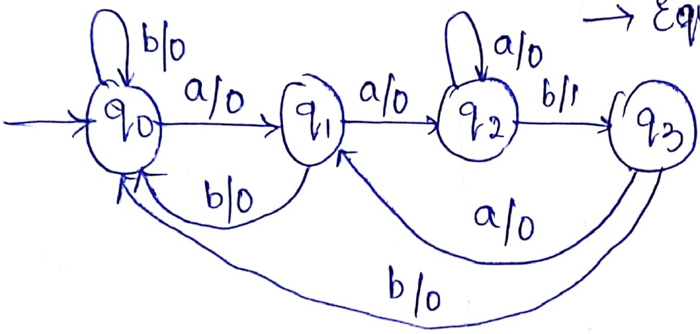
1.

Construct equivalent Meely Machine for the following Moore Machine.



Present State	i/p=a	i/p=b	Output
q_0	q_1	q_0	0
q_1	q_2	q_0	0
q_2	q_2	q_3	0
q_3	q_1	q_0	1
~~q_4~~			

Present state	i/p=a	o/p	i/p=b	o/p
q_0	q_1	0	q_0	0
q_1	q_2	0	q_0	0
q_2	q_2	0	q_3	1
q_3	q_1	0	q_0	0



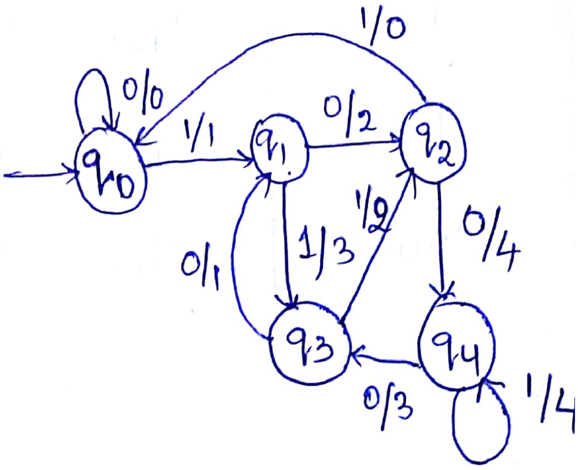
→ Equivalent Meely Machine

2. construct Equivalent Meely Machine for the following Moore Machine.
(Residuo Moduli 5)

Present state	i/p=0	i/p=1	Output
q_0	q_0	q_1	0
q_1	q_2	q_3	1
q_2	q_4	q_0	2
q_3	q_1	q_2	3
q_4	q_3	q_4	4

$0 \div 5 = 0$
$1 \div 5 = 1$
$2 \div 5 =$

Present State	i/p=0	o/p	i/p=1	o/p
q_0	q_0.	0	q_1	1
q_1	q_2	2	q_3	3
q_2	q_4	4	q_0	0
q_3	q_1	1	q_2	2
q_4	q_3	3	q_4	4



3. Construct the Equivalent Moore Machine for the following Meely Machine.

n states m alphabet

~~[illegible]~~.

M+n

Given Meely Machine

Present State	Next state i/p=0	o/p	i/p=1	o/p
q_1	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

$q_1 \rightarrow 1$

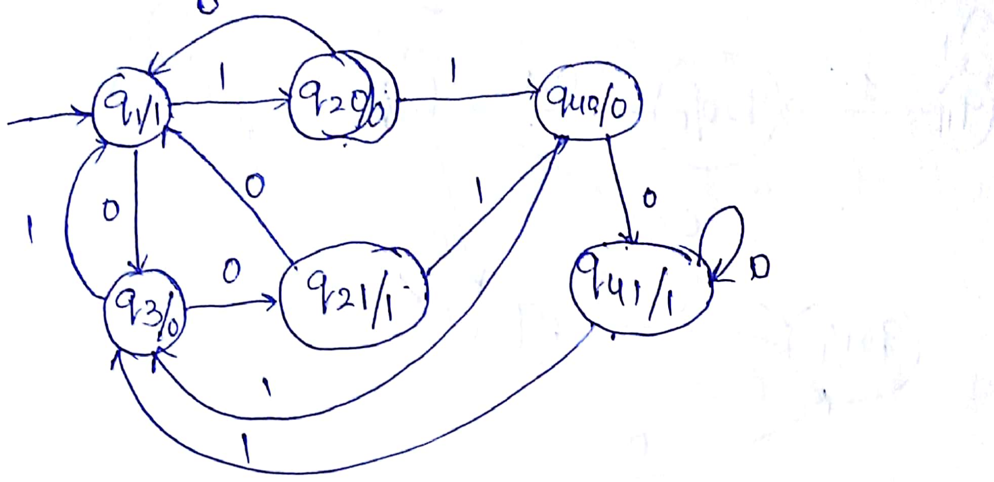
Since q_2 h 2 outputs.

q_{20}

q_{21}

Moore Machine

Present State	i/p=0	i/p=1	Output
q_1	q_3	q_{20}	1
q_{20}	q_1	q_{40}	0
q_{21}	q_1	q_{40}	1
q_3	q_{21}	q_1	0
q_{40}	q_{41}	q_3	0
q_{41}	q_{41}	q_3	1



4. Construct the Equivalent Moore Machine for the following Meely Machine.

Meely Machine:

Present state	i/p=0	o/p	i/p=1	o/p
q_1	q_1	1	q_2	0
q_2	q_4	1	q_4	1
q_3	q_2	1	q_3	1
q_4	q_3	0	q_4	1

Moore Machine:

Present state	i/p =0	i/p=1	Output
q_1	q_1	q_{20}	1
q_{20}	q_4	q_4	0
q_{21}	q_4	q_4	1
q_{30}	q_{21}	q_{31}	0
q_{31}	q_{21}	q_{31}	1
q_4	q_{30}	q_4	1