Toc 1-Lecture

→ one of the most fundamental Courses of CS.

→ will help you to understand how CS as science past 50 years.

→ mainly about what kind of things can you really Compute mechanically, how fast and how much space Does it take to do so

ex Binary Strings that end 0 (zero).

1101101100.

↑ Last bit. = 0 ✓ (accepte).

≠ 0 ✗ (Rejected).

ex Accepts all Valid Java Codes.

↓

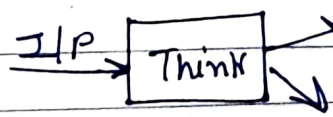
binary. → Valid

→ Invalid.

With help of Compilers we can check.

ex Accepts all Valid Java Codes and never goes into infinite Loop.

→ We can't able to Design Such System that not goes to infinite Loop



Layers.

→ FSM → Simplest Level
↳ having Limited memory

→ CFL → more powerfull than FSM.
↳ higher set of strings

→ Turing machine → high Level
↳ more powerful than above.

Design, or Languages

Undecidable → Those problem which is being not Solved.

Symbol :- a, b, c, 0, 1, 2, 3 - - - -

Alphabet - Σ → Collection of Symbols.
Eg. $\{a, b\}$, $\{0, 1, 2, \}$ - - -

String :- Sequence of Symbols.
Eg. a, b, 0, 1, aa, bb, abi 01 - - - -

Language :- Set of Strings.
Eg. $\Sigma = \{0, 1\}$. from this

L_1 = Set of all Strings of Length 2.
$= \{00, 01, 10, 11\}$.

Finite sets.

L_2 = Set of all Strings of Length 3
$= \{000, 001, 010, 011, 100, 101, 110, 111\}$.

Infinite set: L_3 = Set of all Strings that begin with 0.
$L = \{0, 00, 01, 000, 001, 010, 011, 0000 - - - - \}$.

* Powers of Σ (Sigma):-
$\Sigma = \{0, 1\}$
Σ^0 = Set of all Strings of Length 0.
$\Sigma^0 = \{\in\}$.

Σ^1 = Set of all Strings of Length 1.
$\Sigma^1 = \{0, 1\}$.

Σ^2 = Set of all Strings of Length 2.
$\Sigma^2 = \{00, 01, 10, 11\}$.

Σ^3 = Set of all Strings of Length 3.
$\Sigma^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$.

Σ^n = Set of all strings of Length n.

* Cardinality :- means Num. of elements In Set.

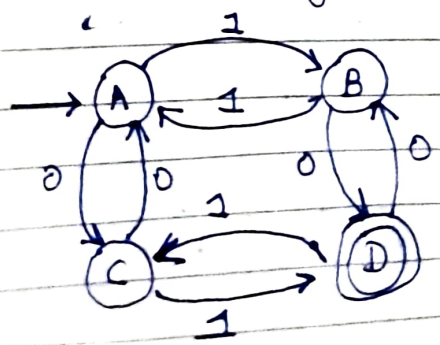
$\Sigma^n = 2^n$ → Cardinality of Σ^n.

* $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \ldots$
$= \{\varepsilon\} \cup \{0,1\} \cup \{00,01,10,11\} \cup \{\ldots\}$.
= Set of all posible strings of all Length over $\{0,1\}$
→ Infinite Set.

Finite state machine

- F.A with o/p
  - Moore machine
  - Mealy machine
- F.A without o/p.
  - DFA
  - NFA
  - ε-NFA

* DFA :- (Deterministic Finite Automata
→ It's the simplest model of Computation.
→ it has very limited memory.



→ Every DFA can be Define as In 5 Tuples.
$(Q, \Sigma, q_0, F, \delta)$.

Q = Set of all States.
Σ = Inputs.
q_0 = Start State / Initial State.
F = Set of final States.
δ = Transition function from $Q \times \Sigma \to Q$.

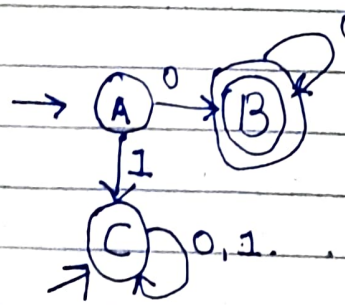
$Q = \{A, B, C, D\}$.
$\Sigma = \{0, 1\}$.
$q_0 = \{A\}$.
$F = \{D\}$.
$\delta =$

	0	1
A	C	B
B	D	A
C	A	D
D	B	C

Transition function. δ.

* Deterministic Finite Automata (Example - 1).

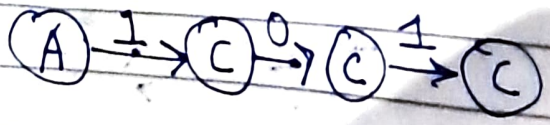
L_1 = Set of all Strings that Start with '0'.
$= \{0, 00, 01, 000, 010, 011, 0000 \ldots\}$.



Eg. 001 → check
Initial State

Dead State / Trap State.

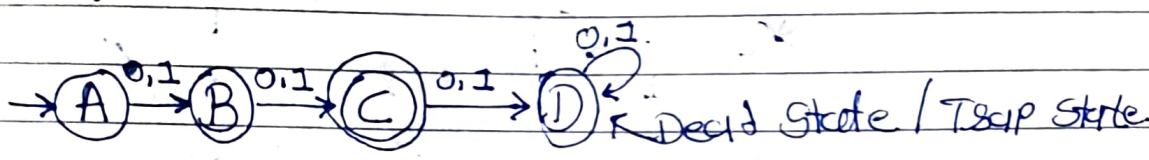
Eg 2. 101.
Initial State.



Ex-2. Construct a DFA that accepts sets of all string over {0,1} of length 2.

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

$L = \{00, 01, 10, 11\}$.

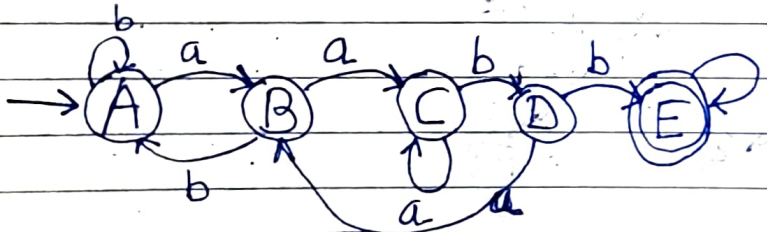


Ex-3 Construct a DFA that accepts any strings over {a,b} that Does not contain the string aabb in it.

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

Try to Design a Simple Problem.

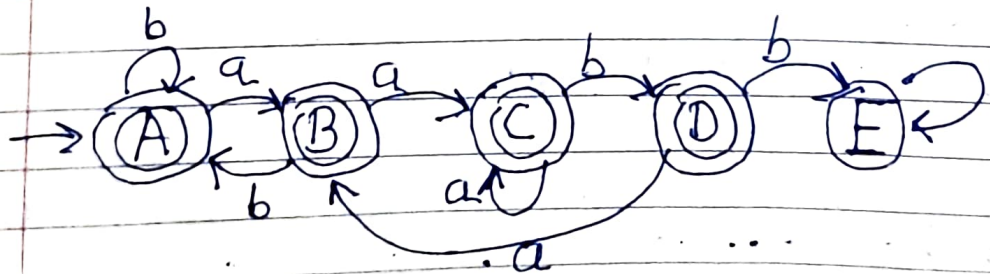
Let us Construct a DFA that accepts all string over {a,b} that Contains the string aabb in it



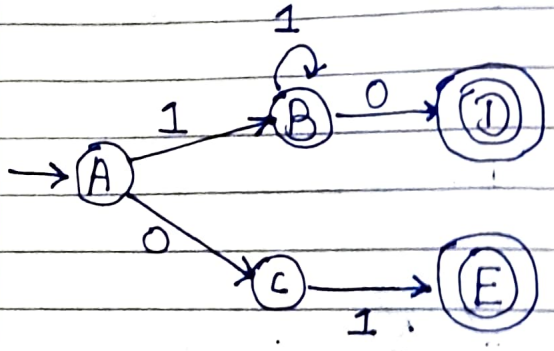
Flip the State → for Doesn't accept aabb.

↳ make the final state into non final state.

↳ make the non final state into final state.



* Ex-4:- How to figure out what a DFA Recognizes



10 ✓

1 1 1 1 1 0 ✓

A B D

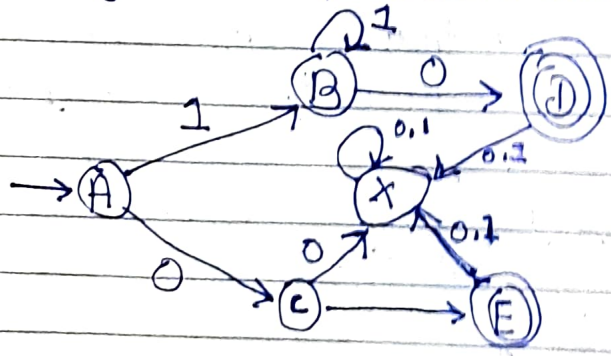
0 1 ✓

E

So we can write it.

L = { Accepts the string 01 OR a string of atleast one '1' followed by a '0' }.

Eg. 001, 010, 011, 1101, 1100.

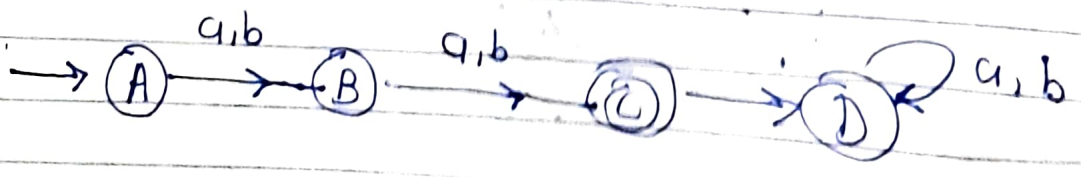


* Ex-5 Construct a DFA over $\Sigma = \{a, b\}$. For the set of all strings of length 2.

L = Set of all strings of length 2.

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

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



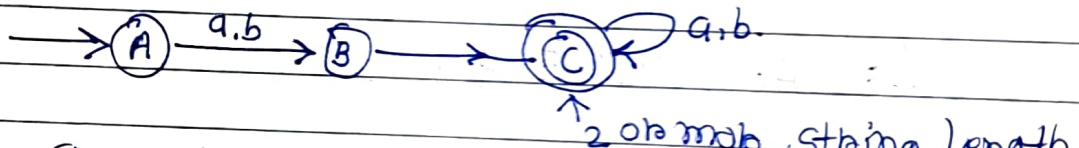
* To finish the DFA's add a self-loop to the last state unless it is an incomplete DFAs.

* Complete DFA - has a transition for all elements of alphabets (Σ) over every state from initial to final.

* Ex-6 :- Construct a DFA for set of all string over $\Sigma = \{a, b\}$ with string length atleast 2.

$\Sigma = \{a, b\}$ $\quad w \geq 2$.

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



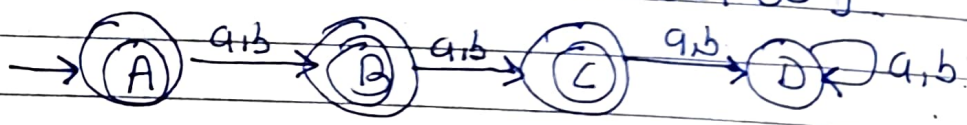
a b a b.

$A \to B \to C \to C \to C$

Ex-7 Construct a DFA of all string $\Sigma = \{a, b\}$ with length at the most 2.

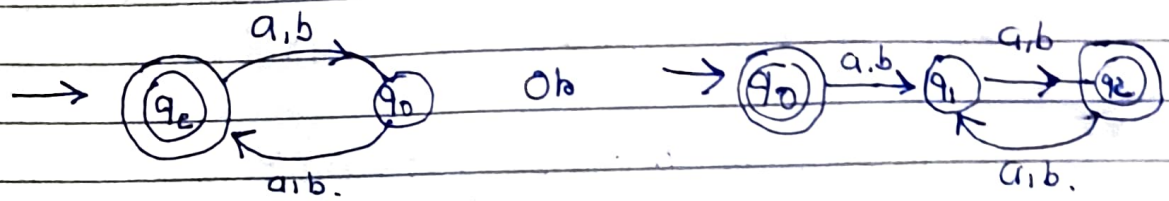
$w \leq 2$.

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



Ex-8 Construct a minimal DFA for set of all strings over $\Sigma = \{a, b\}$ whose string length divided by 2 produces a remainder 0 or string length is even or $|w| \bmod 2 = 0$ or $|w| \cong 0 \bmod 2$.

$L = \{\epsilon, aa, ab, ba, bb, aaaa, bbbb, aaaaaa, bbbbbb \ldots\}$



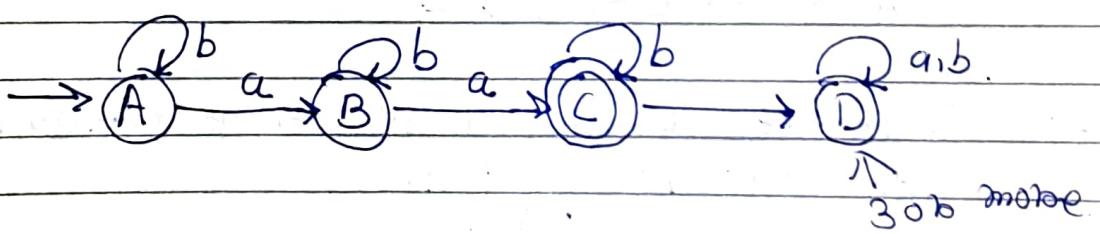
a b a ✗ a b a b ✓

$q_e \to q_0 \to q_e$ $q_0 \to q_1 \to q_2 \to q_1 \to q_2$

Ex-9 Construct a minimal DFA $\Sigma = \{a,b\}$ for a set of all strings in which no. of 'a's are exactly 2.

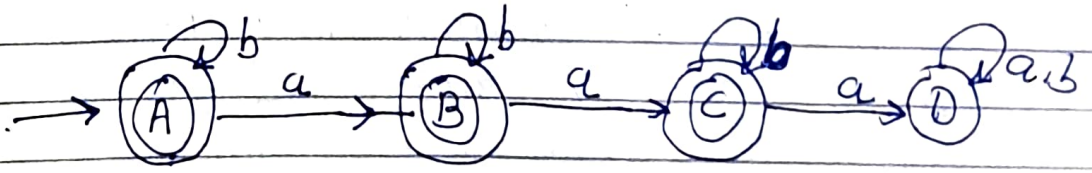
L = DFA : $w \in \{a,b\}$

L = {aa, aba, baa, aab, ababb, abab ---}



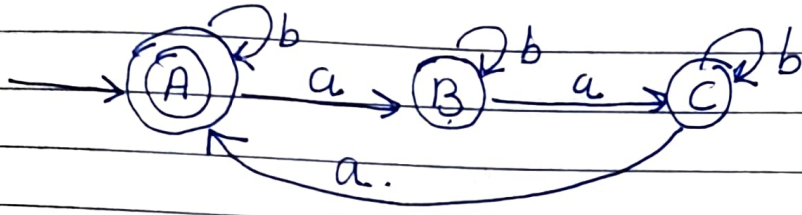
Ex-10 Construct a minimal DFA $\Sigma = \{a,b\}$ for a set of all strings in which no. of 'a's are atmost 2. 'a's = ≤ 2.

L = {∈, a, b, ab, aba, aa --- }.



Ex 11 Construct a DFA over $\Sigma = \{a, b\}$ in which $n_a(w) \bmod 3 = 0$.

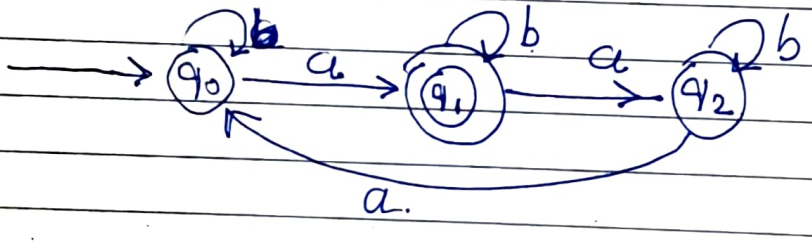
$L = \{\in, aaa, ababa, buuub, abaa, b \ldots\}$.



Ex. 12 Construct a DFA over $\Sigma = \{a, b\}$ in which $n_a(w) \bmod 3 = 1$.

$w \in \{a, b\}$, $n_a(w) \bmod 3 : 1$

$L = \{aaaa, aabaa, baaaa \ldots\}$



TOC / TAFL

Computation : Performing calculations by a machine or calculator.

Mathematical Model – Machine (next TOC)

Symbol : Basic building block for TOC (0-9, A-Z, a-z, !, @, #, ...)

Alphabet : Denoted by Σ
Collection / Set of Symbols

Strings : Set / Collection of alphabets

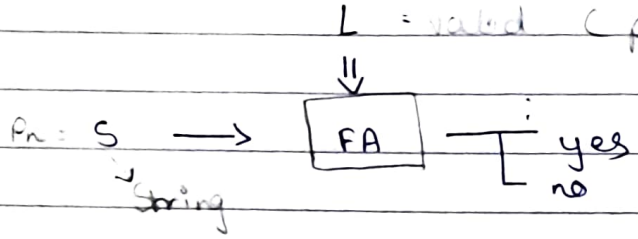
Language :

L_1 : Set of all strings over $\Sigma = \{a, b\}$ of length 2
$L_1 = \{aa, bb, ab, ba\}$ finite set

L_2 : Set of all strings over $\Sigma = \{a, b\}$ of length 3
$L_2 = \{aaa, bbb, aba, aab, abb, baa, bab, bba\}$ finite set

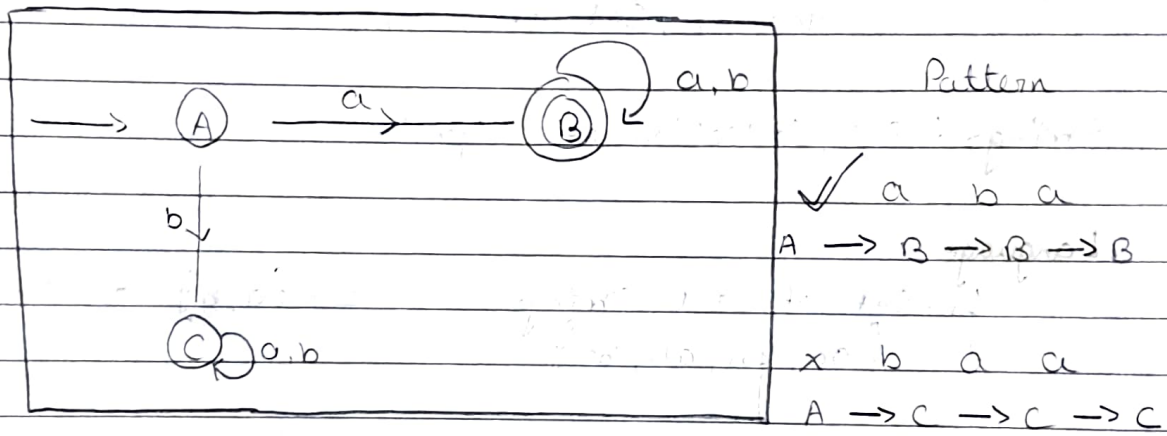
L_3 : Set of all strings over $\Sigma = \{a, b\}$ which starts with 'a'
$L_3 = \{a, aa, ab, aab, aba, \ldots\}$ infinite set

C - program : String
Valid C - program : Language L_1
String $\{P_1, P_2, P_3, P_4, \ldots, P_n\} = L_1$ - infinite set
P_n : String

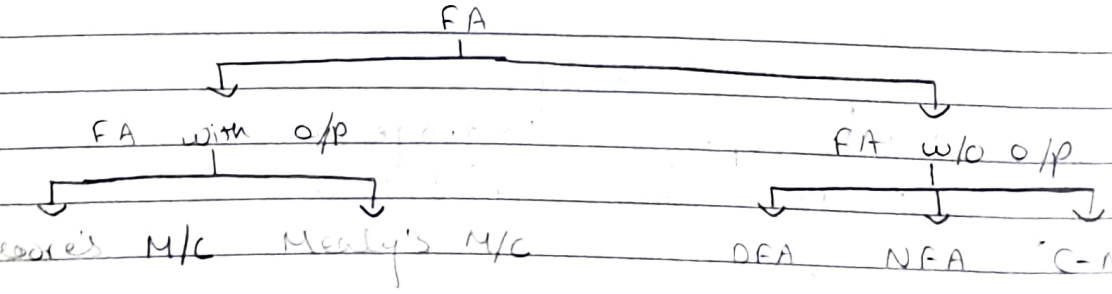


* Infinite set - converted to Finite Representation (known as Finite Automata) - to easily find if there is match or not

Finite Representation :



* A, C → non-final state
* B → Final state
* A → initial



classmate
Date 19/12
Page

DFA :
Deterministic
Finite
Automata
System

quin tuple (5 elements)

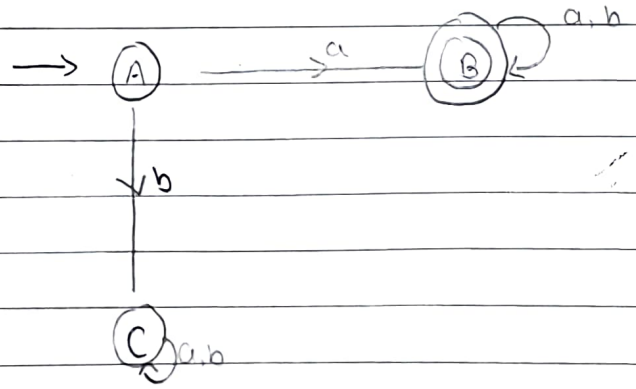
$\delta : Q \times \Sigma \rightarrow Q$

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

Q : set of states
Σ : i/p alphabets
δ : Transition function
q_0 : initial state
F : set of final states

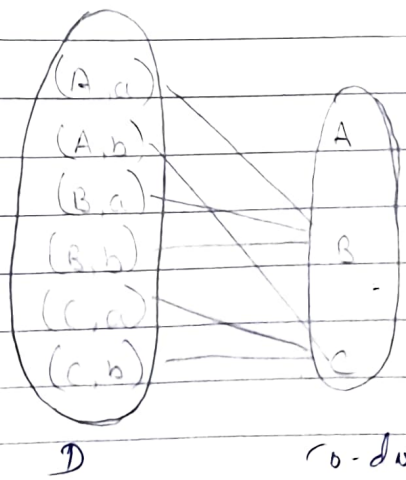
eg : L : set of all strings $\Sigma = \{a, b\}$ which starts with 'a'

$L = \{a, aa, ab, aab, aba, \ldots\}$



$Q = \{A, B, C\}$
$\Sigma = \{a, b\}$
$q_0 = \{A\}$
$F = \{B\}$
$\delta : Q \times \Sigma \rightarrow Q$
domain codomain

$\Rightarrow \{A, B, C\} \times \{a, b\} \rightarrow \{A, B, C\}$



Eg - 1 :

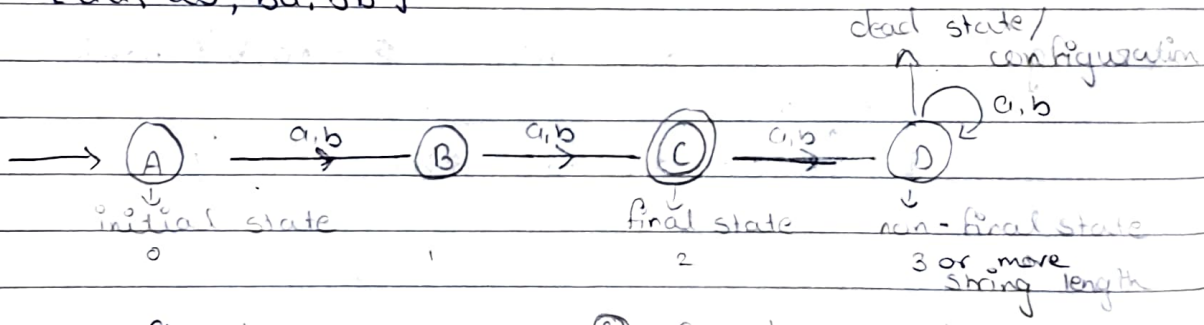
- Construct a DFA over $\Sigma = \{a, b\}$, for the set of all strings of length 2.

L : DFA : set of all strings of length 2

$w = 2$

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

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



① a b

A → B → C - Accepted

② a b a b

A → B → C → D → D Rejected

* To finish the DFAS, add a self-loop to the last state unless it is an incomplete DFAS.

* Complete DFA - has a transition for all elements of alphabet (Σ) over every state from initial to final

Eg - 2 :

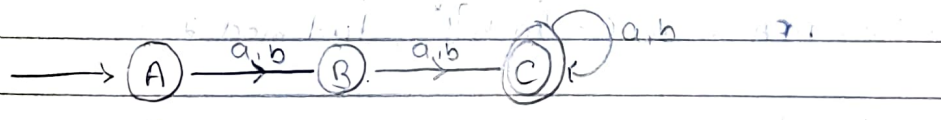
- Construct a DFA for set of all string over $\Sigma = \{a, b\}$ with string length atleast 2.

$w \geq 2$

L : DFA : set of all strings $\Sigma = \{a, b\}$ of length atleast 2

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

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



① a b a b

A → B → C → C → C

Eg - 3 :

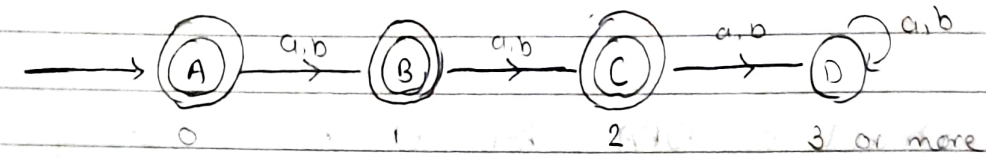
Construct a DFA of all strings $\Sigma = \{a, b\}$ with length at the most 2.

$w \leq 2$.

L : DFA : set of all strings $\Sigma = \{a, b\}$ with string length at most 2

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

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



Eg - 4

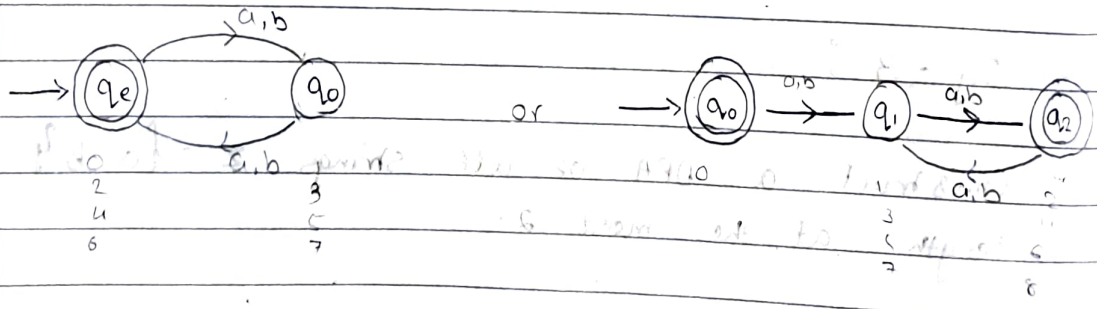
construct with minimal no. of states

Construct a minimal DFA for set of all strings over $\Sigma = \{a, b\}$ whose string length divided by 2 produces a remainder 0 or string length is even or $|w| \bmod 2 = 0$ or $|w| \equiv 0 \bmod 2$.

L = DFA : $w \in \{a,b\}^*$, $|w| \bmod 2 = 0$

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

$L = \{\in, aa, ab, ba, bb, aaaa, bbbb, aaaaaa, bbbbbb, \ldots\}$



① a b a ✗

$q_e \to q_o \to q_e$

② a b a b ✓

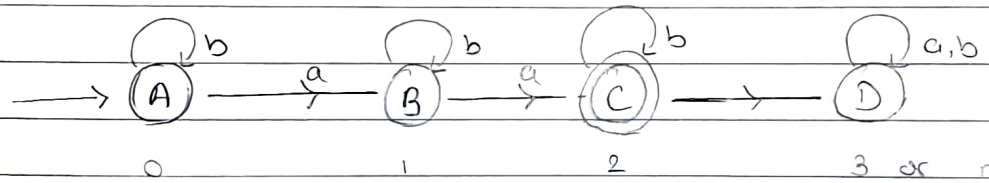
$q_0 \to q_1 \to q_2 \to q_1 \to$

5. Construct a ~~DFA~~ minimal DFA $\Sigma = \{a, b\}$ for set of all strings in which no. of ways 'a's are exactly 2.

L : DFA : $w \in \{a, b\}$, no. of 'a's = 2

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

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



b b a b b a ✓ a b a b a ✗

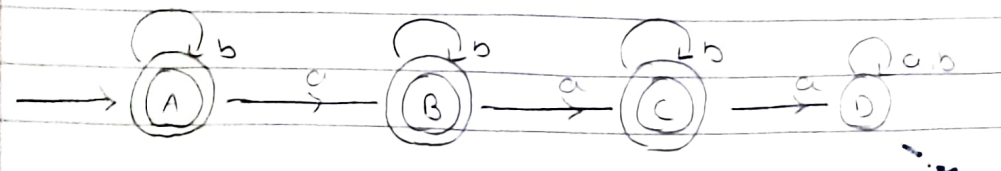
A → A → A → B → B → B → C A → B → B → C → C → C

6 Construct a minimal DFA $\Sigma = \{a, b\}$ for set of all strings in which no. of 'a's are atmost 2.

L : DFA : $w \in \{a, b\}$, no. of 'a's ≤ 2

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

$L = \{\varepsilon, b, a, ab, aba, aa, \ldots\}$

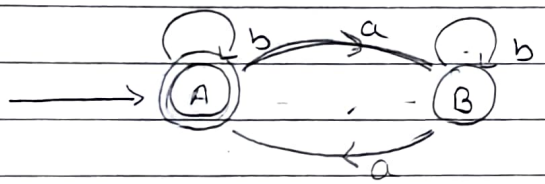


7. Construct a DFA over $\Sigma = \{a, b\}$ for set of all strings in which no. of 'a's is even or $n_a(w) \bmod 2 = 0$ or $n_a(w) \equiv 0 \bmod 2$.

L : DFA : $\omega \in \{a, b\}$, no. of 'a' $n_a(w) \bmod 2 = 0$

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

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

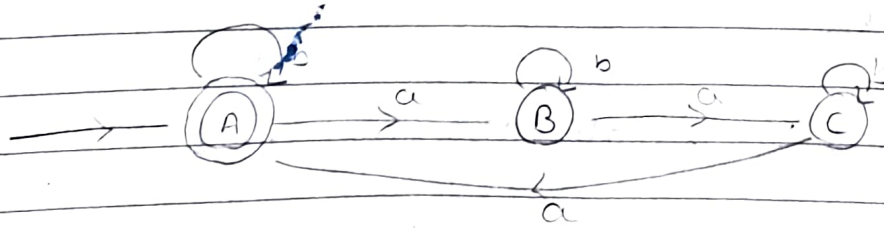


8 Construct a DFA over $\Sigma = \{a, b\}$ in which $n_a(w) \bmod 3 = 0$.

L : DFA : $\omega \in \{a, b\}$, $n_a(w) \bmod 3 = 0$

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

$L = \{\in, aaa, ababa, baaab, abaa, b, \ldots\}$

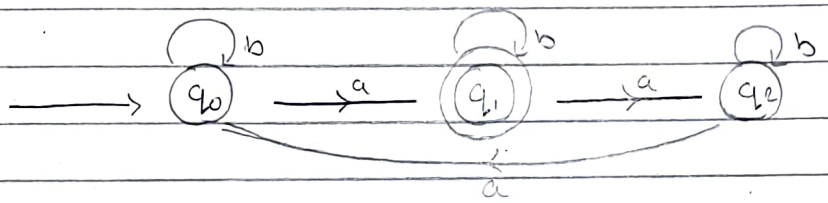


9 Construct a DFA over $\Sigma = \{a, b\}$ in which $n_a(\omega) \bmod 3 = 1$.

L : DFA : $\omega \in \{a, b\}$, $n_a(\omega) \bmod 3 = 1$.

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

$L = \{aaaa, aabaa, baaaa, \ldots\}$



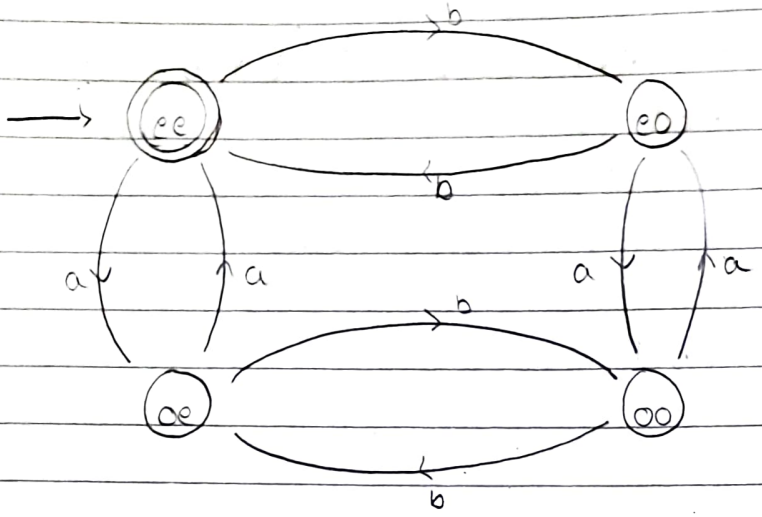
10 Construct a DFA over $\Sigma = \{a, b\}$ in which $n_a(\omega) \cong 0 \bmod 2$ && $n_b(\omega) \cong 0 \bmod 2$.

L : DFA : $\omega \in \{a, b\}$, $n_a(\omega) \bmod 2 = 0$ && $n_b(\omega) \bmod 2 = 0$

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

$L = \{\in, aa, bb, aabb, abab, aaabb, \ldots\}$

a	b	
e	e	$\rightarrow \{\in, aa, bb, aabb, \ldots\}$
e	o	$\rightarrow \{aab, aba, aabbb, \ldots\}$
o	e	$\rightarrow \{abb, bab, aaabb, \ldots\}$
o	o	$\rightarrow \{ab, ba, ababab, \ldots\}$



a a b b ✓

ee → oe → ee → eo → ee

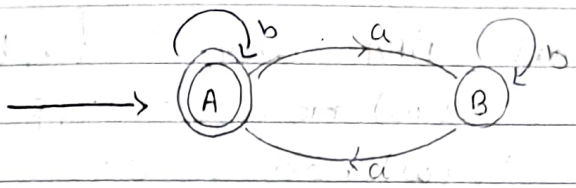
a b a a ✗

ee → oe → oo → eo → oo

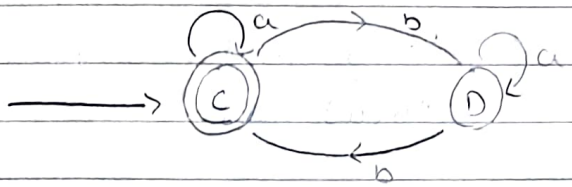
Cross Product Method / Cartesian Product Method :

10 Construct a DFA over Σ = {a, b} in which $n_a(w) \bmod 2 = 0$ & $n_b(w) \bmod 2 = 0$.

Draw separate DFAs of both equations
Make cartesian product of states of both DFAs
Conjoin both DFAs based on cartesian product answers
Final DFA is formed

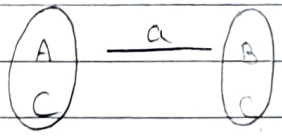


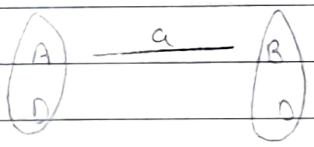
$(n_a(w) \bmod 2 = 0)$

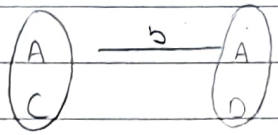


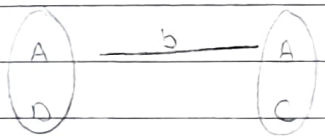
$(n_b(w) \bmod 2 = 0)$

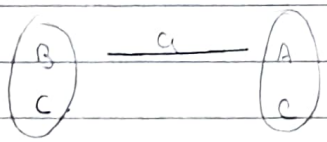
$\{A, B\} \times \{C, D\} = \{AC, AD, BC, BD\}$

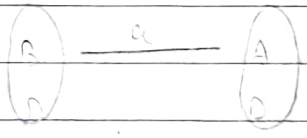


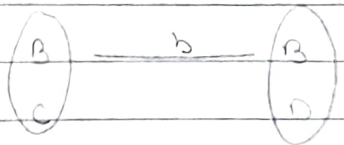


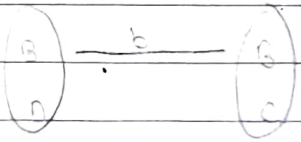


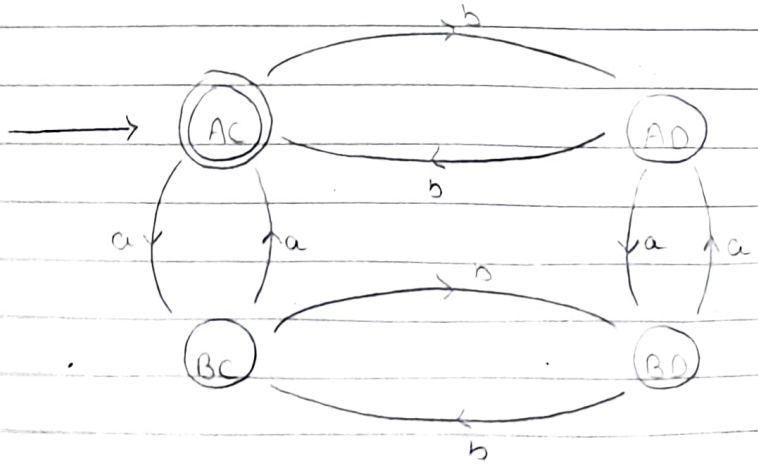










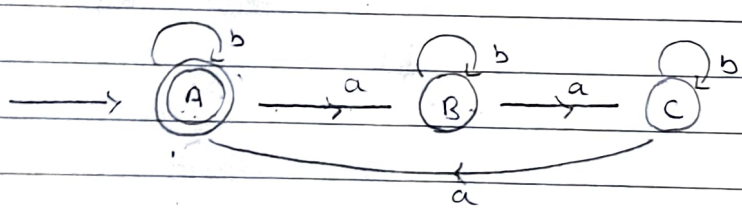


11 Construct a minimal DFA over $\Sigma = \{a, b\}$ for w in which $n_a(w) \bmod 3 = 0$ && $n_b(w) \bmod 2 = 0$.

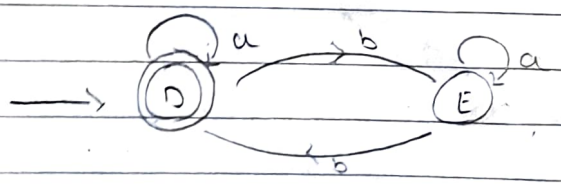
L : DFA : $w \in \Sigma = \{a, b\}$, $n_a(w) \bmod 3 = 0$ && $n_b(w) \bmod 2 = 0$

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

$L = \{\in, aaabb, aaabbbb, baaab, \ldots\}$



($n_a(w) \bmod 3 = 0$)



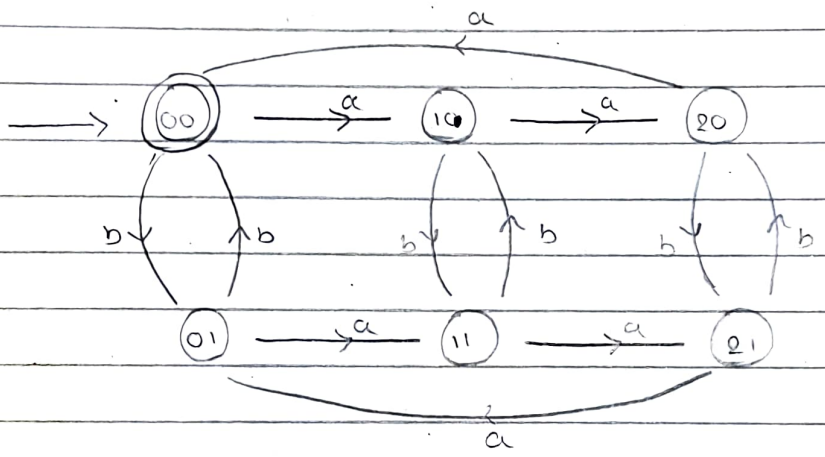
($n_a(w) \bmod 2 = 0$)

$\{A, B, C\} \times \{D, E\} = \underline{AD}, AE, \underline{BD}, BE, \underline{CD}, CE$

	a	b
A	B	A'
D	D	E
B	C	B
D	D	E
C	C	C
D	D	E
A	B	A
E	E	D

B	— a →	C	C	— a →	A	C	— a →	A
E		E	D		D	E		E

B	— b →	B	C	— b →	C	C	— b →	C
E		D	D		E	E		D

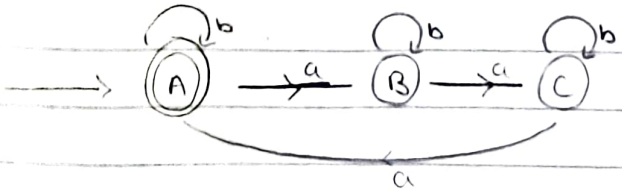


12 Construct a minimal DFA over $\Sigma = \{a, b\}$ for w in which $n_a(w) \bmod 3 = 0$ && $n_b(w) \bmod 3 = 0$.

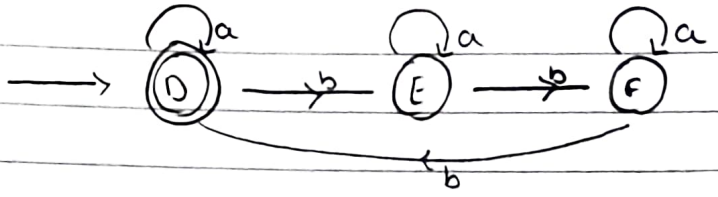
L : DFA : $w \in \Sigma = \{a, b\}$, $n_a(w) \bmod 3 = 0$ && $n_b(w) \bmod 3 = 0$.

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

$L = \{ \in, aaa, bbb, abababab, \ldots \}$

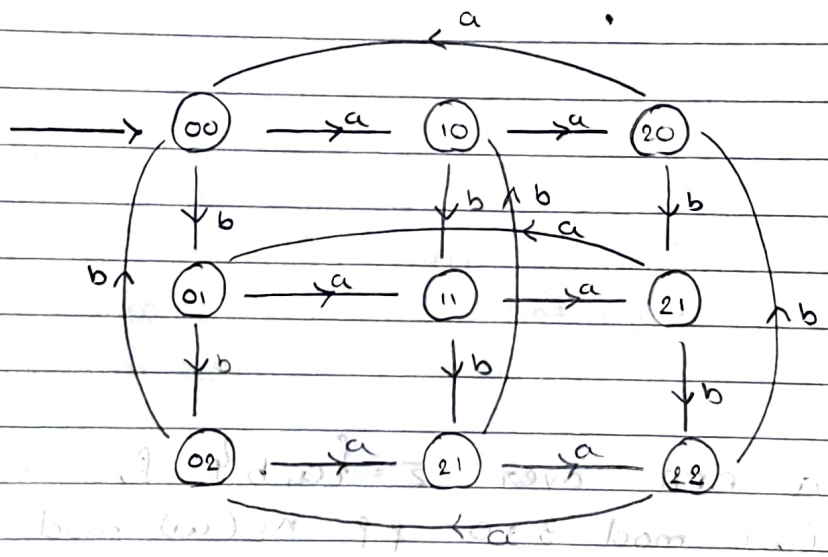


($n_a(w) \bmod 3 = 0$)



$(n_b(w) \bmod 3 = 0)$

$\{A, B, C\} \times \{D, E, F\} = AD, AE, AF, BD, BE, BF, CD, CE, CF$

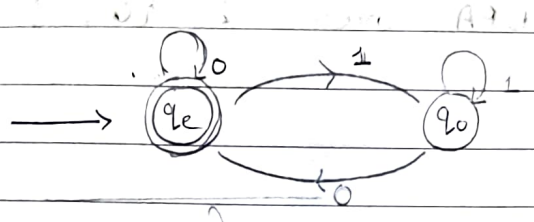


13 Construct a minimal DFA over $\Sigma = \{0, 1\}$ for w in which interpreted as a binary number divisible by 2.

L : DFA : $w \in \Sigma = \{0, 1\}$, $w \bmod 2 = 0$

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

$L = \{0, 10, 110, 110, \ldots\}$



1 0 1 0 ✓

$q_e - q_0 - q_e - q_0 - q_e$

(0) 0000		0001 (1)	
(2) 0010	⅋.2 = 0	0011 (3)	⅋.2 = 1
(4) 0100		0101 (5)	
(6) 0110		0111 (7)	

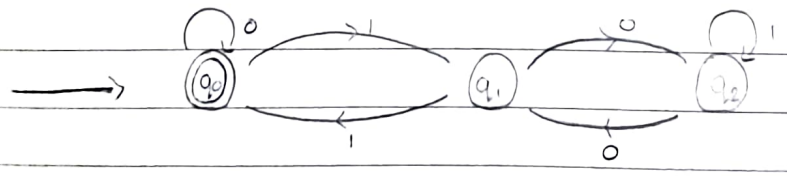
Rough work

14 Construct a minimal DFA M over $\Sigma = \{0, 1\}$ for w ? which interpreted as a binary number divisible by 3. Also construct state transition table.

L : DFA : $w \in \Sigma = \{0, 1\}$, w mod 3 = 0

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

$L = \{0, 11, 110, 1001, \ldots\}$



	0	1
→ q_0^*	q_0	q_1
q_1	q_0	q_2
q_2	q_1	q_2

* : Final state

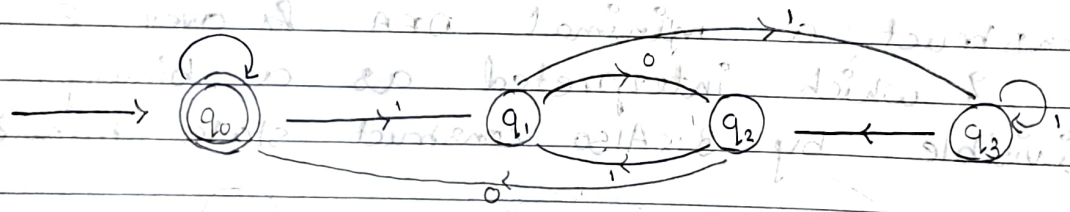
→ : Initial state

15 Construct a minimal DFA over $\Sigma = \{0, 1\}$ for w interpreted as a binary number divisible by 4.

L : DFA : $w \in \Sigma = \{0, 1\}$, $w \bmod 4 = 0$

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

$L = \{0, 0100, 1000, \ldots\}$



	0	1
$\rightarrow q_0^*$	q_0	q_1
q_1	q_2	q_3
q_2	q_0	q_1
q_3	q_2	q_3

To construct this type of state transition table, start with the initial state (q_0) & go sequence-wise via each transition to complete the table.

16 Construct a minimal DFA over $\Sigma = \{0, 1, 2\}$ which are interpreted as ternary number is divisibly by 4.

L : DFA : $w \in \Sigma = \{0, 1, 2\}$, $w \bmod 4 = 0$

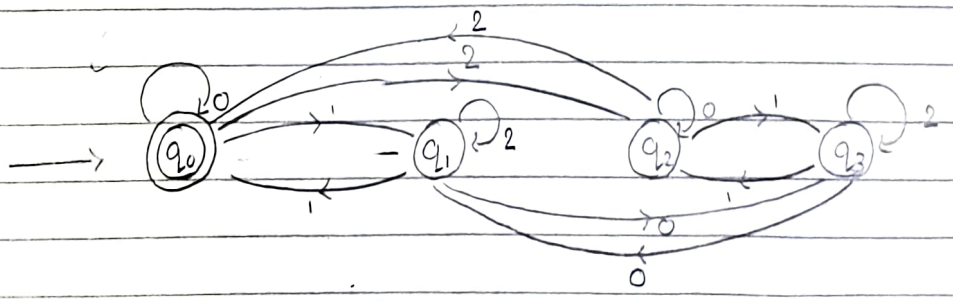
$\Sigma = \{0, 1, 2\}$

$L = \{0, 11, 22, \ldots\}$

$(11)_3 = (1 \times 3) + 1 = 3 + 1 = 4$

$(22)_3 = (2 \times 3) + 2 = 6 + 2 = 8$

	0	1	2
$\rightarrow q_0^*$	q_0	q_1	q_2
q_1	q_3	q_0	q_1
q_2	q_2	q_3	q_0
q_3	q_1	q_2	q_3

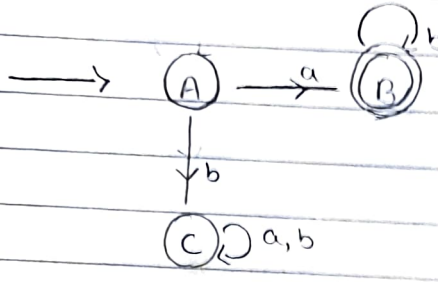


17 Construct a minimal DFA over $\Sigma = \{a, b\}$ for set of all strings starting with 'a'.

L : DFA : $w \in \Sigma = \{a, b\}$, w starts with 'a'

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

L = { 'a', ab, abb, aba, ... }

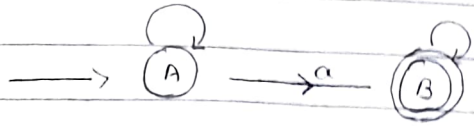


18 Construct a minimal DFA over $\Sigma = \{a, b\}$ for set of all strings containing 'a'.

L : DFA : $w \in \Sigma = \{a, b\}$, $a \in w$

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

L = { a, ab, ba, baba, aaba, ... }



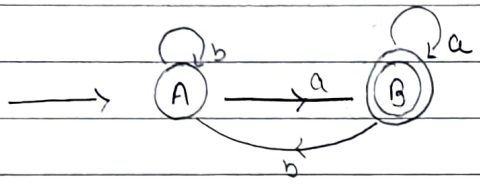
19 Construct a minimal DFA over $\Sigma = \{a, b\}$ for set of all strings ending with 'a'.

L :

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

$L = \{a, ba, aa, aaba, \ldots\}$

	a	b
→ A	B	A
B	B	A

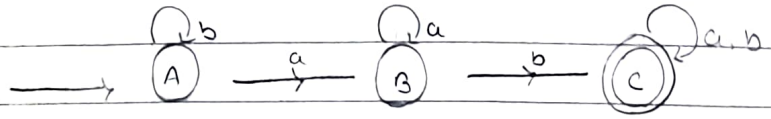


20 Construct a DFA over $\Sigma = \{a, b\}$ for set of all strings which contain 'ab'.

L : DFA : $w \in \Sigma = \{a, b\}$, 'ab' $\in w$

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

$L = \{ab, aab, bab, abab, baba \ldots\}$



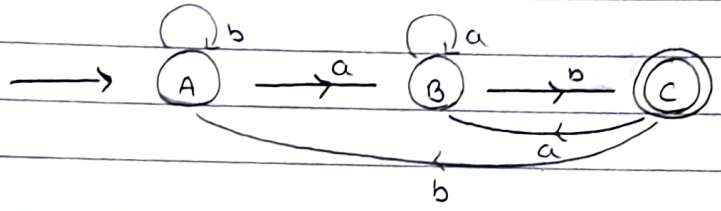
	a	b
→ A*	B	A
B	B	C
C	A C	C

21. Construct a DFA over $\Sigma = \{a, b\}$ for set of all strings ending with 'ab'.

L : DFA : $w \in \Sigma = \{a, b\}$, w ends with 'ab'

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

$L = \{ab, bab, aab, abab, \ldots\}$



22. Construct a DFA over $\Sigma = \{a, b\}$ for set of all strings that starts with 'a' & ends with 'b'.

L : DFA : $w \in \Sigma = \{a, b\}$, w starts with 'a' & ends with 'b'

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

$L = \{ab, aab, abab, aaab, \ldots\}$