FLAT

→ Introduction:-

FLAT/TOC:- Formal Languages & Automata Theory

Applications:-

1. Lexical Analysis
2. String Matching Algorithms
3. Network protocols
4. Compilation of regular expressions
5. Analysis of boolean program
6. To design Text editors
7. To design spell checkers
8. Sequential circuits design

→ Formal Language:-

The language which has proper set of Alphabet, Grammer rules and a model to recognize is called as Formal Language.

→ Automata:-

It is a model/abstract model of machine that perform computation on input by moving through a series of states

→ Automation:- Denotes Automatic process carrying out the produ-ction of process

FLAT:- Deals with logic of computation with respect to simple machine.

Formal Languages

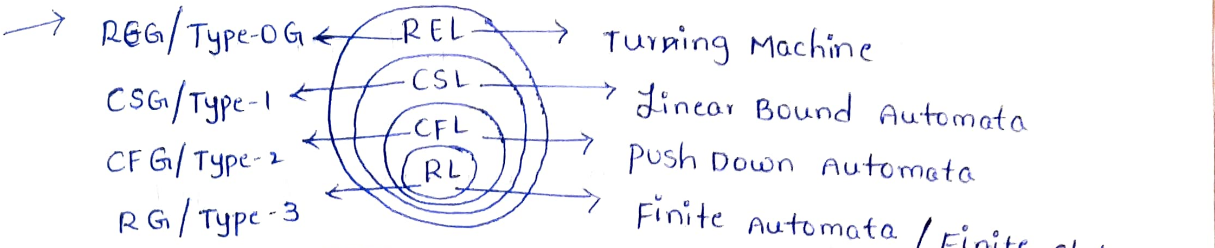
→ FL divides into 4 categories

1. Regular Language → It has Regular Grammer
2. Context Free Language → Context Free Grammer
3. Context Sensitive Language → Context Sensitive Grammer
4. Recursive Enummerable Language → REGrammer/Unrestricted Grammer

→ The set of rules is "Grammer"

Formal Languages

	Regular Language (RL)	CFL	CSL	REL
	↓	↓	↓	↓
Grammer →	Regular Grammer (RG)	CFG	CSG	REG / Unrestricted Grammer
	↓	↓	↓	↓
Automata →	Finite Automata (FA)	Push Down Automata (PDA)	Linear Bound Automata (LBA)	Turing Machine (TM)



$\boxed{RL \subset CFL \subset CSL \subset REL}$

① **Alphabets :-** The non-empty finite set of symbols is known as Alphabets and it is denoted by the symbol Σ

Ex:-

① Binary Language $\Sigma = \{0, 1\}$

Decimal Language $\Sigma = \{0, 1, 2, \dots, 9\}$

English " $\Sigma = \{A, B, \dots Z\} / \{a, b, \dots z\}$

② **String :-** The sequence of symbols & choosen from the Alphabets Σ, is known as string and it is denoted by W.

Ex:- $\Sigma = \{0, 1\}$; $W_1 = 010$, $W_2 = 1010$, $W_3 = 10101001 \dots$

$\Sigma = \{A, B, \dots Z\}$; $W_1 = FLAT$, $W_2 = IIIT$, $W_3 = XYZX \dots$

$\Sigma = \{0, 1, \dots 9\}$; $W_1 = 937$, $W_2 = 01234$, $W_3 = 7845 \dots$

→ **Length of a string :-** If W is any string defined over the alphabet Σ then Number of symbols involved in the string is known as Length of a string. Denoted as $|W|$

Ex: $W = 0101$, $W_2 = RGUKT$, $W_3 = 489673$

$|W| = 4$, $|W_2| = 5$, $|W_3| = 6$

→ Empty String :- If a string having length '0' is Empty St..

(or)

A string with zero occurrences of symbols is also called Empty string.

→ Denoted by epsilon ϵ

Note:- ① If W is any string then $|W| \geq 0$ ($|W| = 0$ (or) $|W| > 0$)

↓ Empty string ↓ Non-Empty String

If $W = \epsilon \rightarrow |W| = 0$

$W \neq \epsilon \rightarrow |W| > 0$ (or) $|W| \neq 0$

② Contatenation

$W \cdot \epsilon = W = \epsilon \cdot W \Rightarrow$ Contcatenation

Ex :- $010 \cdot \epsilon = 010$

③ $\epsilon \cdot \epsilon = \epsilon$

④ Let u, v are 2 strings and $|u|, |v|$ are length

$$\boxed{|uv| \leq |u| + |v|}$$

→ Sub String :- Let u, w be the 2 strings define over Σ

then u is said to be substring of 'W'

if 'u' occurs in 'W'.

Ex :- $W = CSE$

$u = C, S, E, CS, SE, EC, ES, CE, CSE$

Ex2 :- $W = FLAT$

$u = \epsilon, F, L, A, T, FL, LA, AT, FLA, LAT, FLAT$

Note:-

1. Every string is a substring for itself
2. Empty string (ϵ) is a substring for everything
3. If u is substring of 'w' then $|u| \le |W|$

Types of substrings

1. Trivial substrings / Improper substrings
2. Non-Trivial substrings / proper substrings

Trivial Substring:-

If 'W' is any string defined over Σ then the substring 'W' itself and empty string 'ϵ' are called as Trivial substrings.

$$T.S = \{ \epsilon, W \}$$

Non-Trivial Substring:-

If 'W' is any string defined over Σ then every substrings of W other than 'W' itself and empty string 'ϵ' is known as Non-Trivial substrings.

W = CSE

Trivial substrings = ϵ, CSE

Non - Trivial = C, S, E, CS, SE

Ex:-

W = CSE

$l=0 \Rightarrow \epsilon$ - 1 - T

$l=1 \Rightarrow$ C, S, E - 3 } N.T

$l=2 \Rightarrow$ CS, SE - 2

$l=3 \Rightarrow$ CSE - 1 - T

W = FLAT

$l=0 \Rightarrow \epsilon$ - 1 - T

$l=1 \Rightarrow$ F, L, A, T - 4 }

$l=2 \Rightarrow$ FL, LA, AT - 3 } N.T

$l=3 \Rightarrow$ FLA, LAT - 2 }

$l=4 \Rightarrow$ FLAT - 1 - T

Note:-

→ If 'W' is any string defined over a Σ and $|W|=n$ (assuming all the symbols are distinct)

→ Number of substrings $= \frac{n(n+1)}{2} + 1 = \Sigma n + 1$

→ Number of Trivial substrings $= 2$

→ Number of non-Trivial substrings $= \Sigma n - 1$

Prefix:- The sequence of leading or starting symbols

Ex:- $W = CSE$

prefix:- C, CS, ~~SE~~, CSE, ϵ

Suffix:- The sequence of Ending or tailing symbols

suffix:- ϵ, CSE, E, SE

Note:-

① Trivial substrings of W acts as both prefixes and suffixes.

② If $|W|=n$ then

No. of prefixes = No. of suffixes = $n+1$

② **Language:-** Collection of strings defined over Σ

Ex:- $\Sigma = \{0,1\}$

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

$L = \{W \in \{0,1\}^* / |W| = 3\} \Rightarrow 000, 001, 010, 011, 100, 101, 110, 111$

$L = \{0^n 1^n / n \geq 1\} \Rightarrow 0^1 1^1, 0^2 1^2, 0^3 1^3 \ldots$

$01, 0011, 000111$

$L = \{0^n 1^n 0^n / n \geq 1\} \Rightarrow 010, 0011, 00, 000111000, \ldots$

$L = \{0^n 1^m / m \neq n\} \Rightarrow 011, 001, 0001, 00011, \ldots$

$L = \{0^m 1^n / m < n\} \Rightarrow 011, 00111, 0111, 01111 \ldots$

Power of Alphabets

If Σ is any alphabet then Σ^k is the set of all strings over the alphabet Σ of length exactly 'k'

$$\Sigma^k = \{W \in \{0,1\}^* / |W| = k\}$$

Ex:- $\Sigma = \{0, 1\}$

$\Sigma^0 = (l=0) \Rightarrow \epsilon \quad - 2^0$

$\Sigma^1 = (l=1) \Rightarrow 0, 1 \quad - 2^1$

$\Sigma^2 = (l=2) \Rightarrow 00, 01, 10, 11 \quad - 2^2$

$\Sigma^3 = (l=3) \Rightarrow 000, 001 \ldots \ldots 111 - 2^3$

$\Sigma^4 = (l=4) \Rightarrow 0000, \ldots \ldots 1111 - 2^4$

$\Sigma^5 = (l=5) \Rightarrow 00000, \ldots \ldots 1111 - 2^5$

$\vdots$

$\Sigma^k = (l=k) \Rightarrow \quad - 2^k$

Note:-

→ '*' ⇒ kleen closure   '+' ⇒ positive closure

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

$$= \bigcup_{i=0}^{\infty} \Sigma^i = \{W \in \Sigma^* / |W| \geq 0\}$$

② $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^4 \cup \ldots$

$= \cancel{\sum_{i=1}^{\infty}} \bigcup_{i=1}^{\infty} \Sigma^i = \{ w \in \epsilon^+ / |w| \geq 1 \}$

→ $\Sigma^* = \{ \epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots \}$

→ $\Sigma^+ = \{ 0, 1, 00, 01, 10, 11, 000, \ldots \}$

Note :-

① Σ^* is called as Universal language

② $\Sigma^+ \subset \Sigma^*$

③ $\Sigma^* = \Sigma^+ \cup \{\epsilon\}$

④ $\Sigma^+ = \Sigma^* - \{\epsilon\}$

⑤ $\Sigma^* \cup \Sigma^+ = \Sigma^*$

⑥ $\Sigma^* \cap \Sigma^+ = \Sigma^+$

→ If 'L' is any language defined over the alphabet Σ then 'L' is always a part of Σ^*

Ex:-

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

$\Sigma^0 = \epsilon$

$\Sigma^1 = 0, 1$

$\Sigma^2 = 00, 01, 10, 11$

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

$\Sigma^* = \bigcup_{i=0}^{\infty} \Sigma^i$

→ Set :- Collection of strings

- Set
  - Empty set (set with 0 elements)
  - Non-Empty set
    - Finite set (Elements are countable)
    - Infinite set (Elements are uncountable)
      - Countable set
      - Uncountable set

→ Language :- Set of strings

- Language
  - Empty Language (Language with 0 elements)
  - Non - Empty Language
    - Finite Language
    - Infinite Language

→ Empty Language :- The language which doesn't contain even a empty string 'ϵ' is called as EL

L is empty language $\Leftrightarrow$ $|L| = 0$ $\Rightarrow$ $L = \{\}$

$|L| = 0$

→ Non- Empty Language :-

The language "L" which contains atleast 1 string is called as Non-Empty language.

L is non- empty $\Leftrightarrow$ $|L| \neq 0$

Ex:- $L = \{0, 1\}$ $\Rightarrow$ $|L| = 2$

$L = \{\epsilon\}$ $\Rightarrow$ $|L| = 1$

$L = \{0^n / 1 < n \le 10\} \Rightarrow |L| = 9 \rightarrow$ Finite

$L = \{0^n / n \ge 1\} \Rightarrow |L| = \infty$

$L = \{0^m 0^n / m > n\} \Rightarrow |L| = \infty$

$\}$ Infinite

→ **Finite Language :-**

The language contain finite no of strings then length of each and every string is finite. is known as Finite Language.

Ex :-

$L = \{01, 10\}$

$L = \{0, 1, 00, 100, 101, 1010\}$

$L = \{\Sigma^n / n = 100\}$ $\Sigma = \{0, 1\} \Rightarrow$ {Length} = 100

2^{100} strings

$L = \{\Sigma^n / 0 \le n \le 10\}$ $\bigcup_{i=1}^{10} \Sigma^i$

→ **Infinite Language :-**

The Language which contain Infinite no of strings where the length of each and every string is finite is called Infinite Language

Ex :-

$L = \{0^n / n \ge 1\}$

$L = \{0^m 0^n / m = n\}$

$L = \{0^m 1^n / m > n\}$

→ Empty Language ⇒ Can be a RL, CFL, CSL, REL

→ Finite ~~and Infi~~ ⇒ RL, CFL, CSL, REL

→ Infinite - RL (NRL)

CFL	NCFL
CSL	NCSL
REL	NREL

Note:

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

→ power set $(\Sigma) = \{\epsilon, \{0\}, \{1\}, \{0,1\}\}$

→ Σ = alphabet $\{0,1\}$

Σ^* = set of all strings defined over Σ

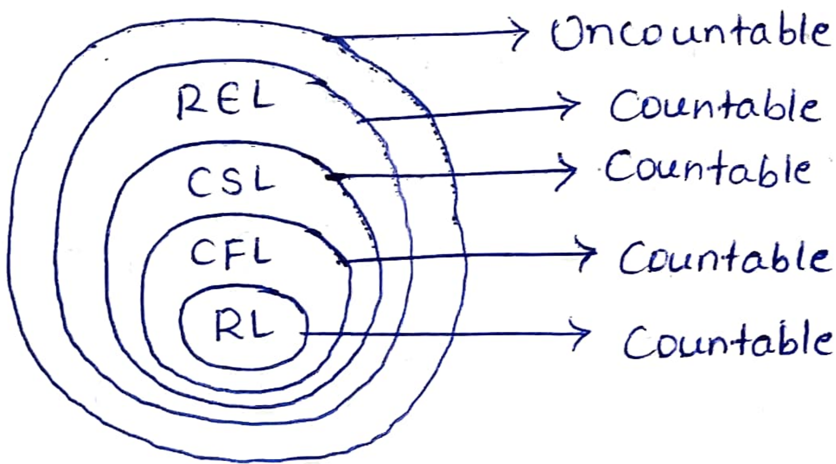
$\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, 111, \ldots\}$

↳ No. of strings is uncountable

power set (Σ^*) = Uncountable set

No. of elements in power set $= 2^{\Sigma^*}$

→



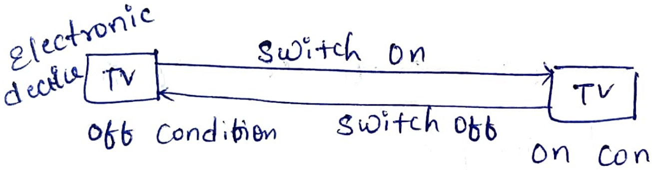
→ Finite Automata / Finite State machine / FSA

Automata :- It is a abstract model of a machine that perform computations on input by moving through a series of states.

FSA :- Applications

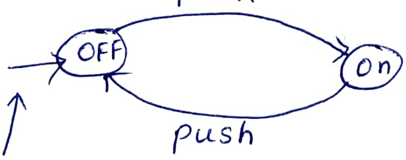
1. Lexical Analysis - Tokenization (can be acheived by using FSA)
2. Text editing
3. Computer Networking

ex: → You are having a TV



→ In order to off and on, we should perform some actions

→ Finite Automate modelling is an on/off switch



Arrow mark indicates initial state

off and on different states Transitions

→ Finite state Machine

states :- states are represented by circles

Start state

Final state

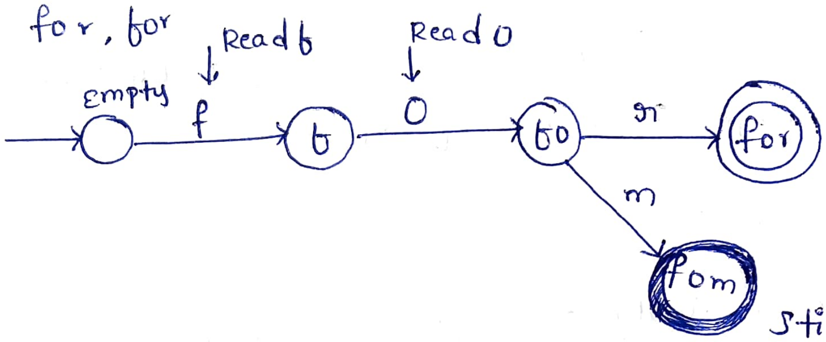
One (or) more FS

→ Also called Accepting states

 → Two circles

Trasition :- Move from one state to other state

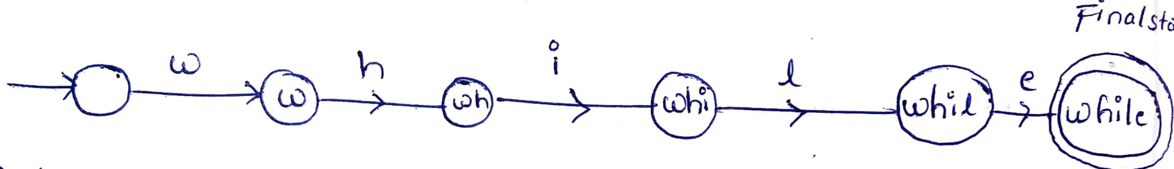
Ex :- for, for



keyword in c library

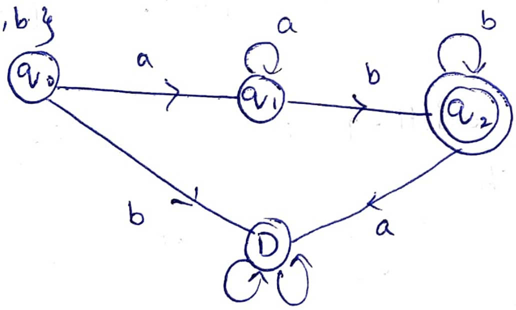
Still continuation is there and not a keyword.

Ex2 :- while



→ If a word is in final state then it is a keyword in c library, by Lexical analysis.

Ex :- $\Sigma = \{a, b\}$



Let us consider

→ 4 states :- q_0, q_1, q_2, D

→ q_0 → Initial state

q_2 → Final state

D → Dead state

Input ① aabb - ✓ Accepted

$$q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{b} q_2$$

② baab

③ abab

④ ab

⑤ aabba

→ After reading the string if it is Final state then it is Accepted by FA.

→ Block diagram of FA

→ FSM Consists three parts

① Input tape

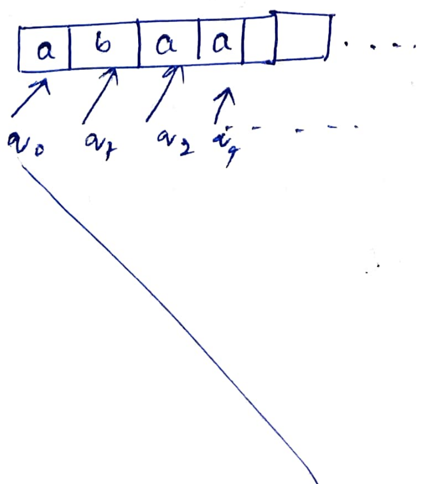
② Tape Header

③ Finite Control

Input tape:-

Input tape is divided into cells and each cell has capability to store one symbol at any point of time.

→ The Input tape contains only one Input state



→ Tape Header (↑)

→ Tape Header reads the input symbol from the input tape starting from the left most end.

→ At any point of time the Tape Header points to only one input symbol.

→ After taking input symbol from input tape. The Tape Header moves to exactly one cell a header towards right side.

∴ The movement of TH in FA is unidirectional (Left to Right)

→ Because of this unidirection, the FA can be used for scanning of the strings.

→ Finite Control:

→ Finite control manages the transitions of FA i.e whether it is moving to right side after taking this input signal or not,

→ List of system state } These are all taken care by FA

→ In FA, At regular intervals the TH reads one symbol from input and then enters a new state.

→ The new state depends only on the current and the symbol just read.

→ Acceptance by FA :-

Let W be any string defined over Σ

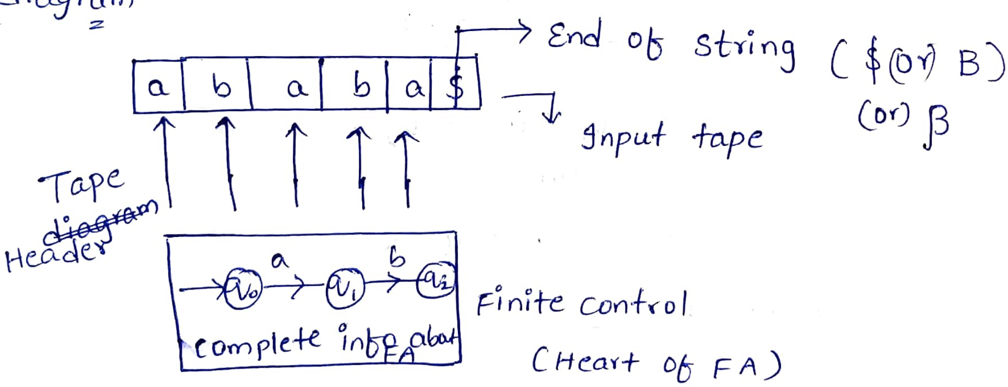
→ Corresponding to W, If there existed transition path which starts at initial state and ends in any one of the Final state.

→ The string W is accepted by FA / FSM

⇒ The set of all string which are accepted by FA is called as Language of FA.

$$L(FA) = \{ W / \delta(q_0, W) = FS \}$$

→ Block diagram



The state of FA

↓

a

Now state moves to b (Tape Header)

↓

b

Formal Definition of Finite Automata

→ A Finite state Automation M is a five type (Quintuple) variable

$$M = \{Q, \Sigma, \delta, q_0, F\}$$ where

Q is a finite set of states

Σ is a finite set of input symbols / Alphabet

q_0 in Q is the Initial state of M

$F \subseteq Q$ is the set of Final states

δ is mapping from $Q \times \Sigma$ into Q (Transition function)

$$\delta : \underbrace{Q}_{\text{set of states}} \times \underbrace{\Sigma}_{\text{Alphabet}} \rightarrow Q \rightarrow \text{set of states}$$

Ex:- $q_1 \xrightarrow{0} q_2$

$\delta : q_1 \times 0 \rightarrow q_2$ ✓

(or)

$\delta(q_1, 0) = q_2$ ✓

From this Ex:-

$$\delta : \underbrace{Q}_{\text{Current of FA}} \times \underbrace{\Sigma}_{\text{Current State of}} \rightarrow \underbrace{Q}_{\text{next state of FA}}$$

→ Representation of FA

1. Transition Diagram
2. Transition Table

→ Trasition Diagram

→ Transition Diagram (TD) for a FA is $M = \{Q, \Sigma, \delta, q_0, F\}$ is a graph as follows

① For each state in Q there is a node q_0 q_1

② If there is mapping $\delta(q, a) = p$, Then TD must have an Arc from node q to Node p labeled with a.

$$q \xrightarrow{a} p$$

③ There must be an arrow into start state q_0, labele as start.

$$\xrightarrow{\text{start}} q_0$$

④ Nodes with respect to accepting states (Final states defined are maked by double circles.

$$q_1 \longrightarrow q_2$$

Ex:-

TD

$M = \{Q, \Sigma, \delta, q_0, F\}$

$Q = \{q_0, q_1, q_2\}$ $\Sigma = \{a, b\}$

$q_0 = \{q_0\}$ δ: $\delta(q_0, a) = q_0$ $F = \{q_2\}$

$\delta(q_0, b) = q_1$

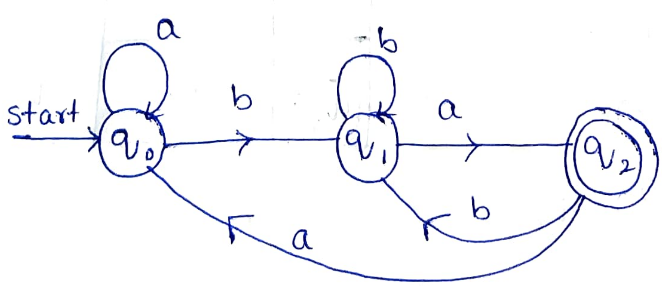
$\delta(q_1, a) = q_2$

$\delta(q_1, b) = q_1$

$\delta(q_2, a) = q_0$

$\delta(q_2, b) = q_1$

FA



→ Transition Table

→ TT is a convential tabular representation of a function(TF) like δ that takes 2 arguments (CS, Input Symbol) and returns a value Next State

① Row of a table corresponds to states in Q

② Columns of a table corresponds to input symbol

③ The entry for the row w.r.t state q and the column with respect to input a is the state $\delta(q, a)$

④ Start state marked with an arrow →

⑤ Final state marked with $*$ or circle

$* q_f$ or q_f (circled)

TT :

$M = \{Q, \Sigma, q_0, \delta, F\}$

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

$\Sigma = \{a, b\}$, $q_0 = \{q_0\}$, $F = \{q_2\}$

δ : $\delta(q_0, a) = q_0$

$\delta(q_0, b) = q_1$

$\delta(q_1, a) = q_2$

$\delta(q_2, a) = q_0$

$\delta(q, b) = q_1$

$\delta(q_2, b) = q_1$

I/P Symbol / States	a	b
$\rightarrow q_0$	$\delta(q_0, a) = q_0$	$\delta(q_0, b) = q_1$
q_1	q_2	q_1
q_2	q_0	q_1

Assignment

① $M = \{Q, \Sigma, q_0, \delta, F\}$

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

$q_0 = \{q_0\}$, $F = \{q_2\}$, $\Sigma = \{a, b\}$

δ :

$\delta(q_0, a) = q_0$

$\delta(q_0, b) = q_1$

$\delta(q_1, a) = q_1$

$\delta(q_1, b) = q_2$

$\delta(q_2, a) = q_1$

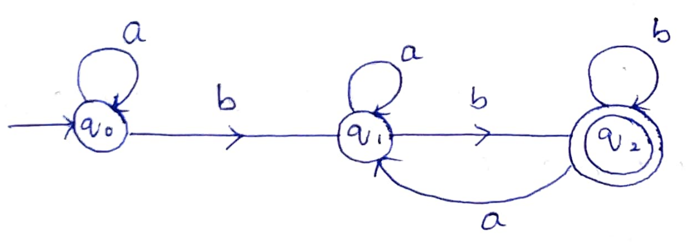
$\delta(q_2, b) = q_2$

a) Construct the TD

ⓑ construct the TT

c) prepare a list of strings accepted by FA (L(FA))

TD



TT

	a	b
→ q_0	q_0	q_1
q_1	q_1	q_2
q_2	q_1	q_2

Language of FA

① ababb

② abb

③ abab

Finite Automata

Finite Automata without output
- Deterministic Finite Automata (DFA)
- Non-Deterministic Finite-Aotomata (NFA)

Finite Automata with output
- Moore Machine
- Melay Machine

→ Deteministic Finite Automata (DFA):-

The mathematical model of DFA M is defined as 5 tuple variable (Quintuple)

$$M = \{Q, \Sigma, \delta, q_0, F\} \text{ where}$$

Q is a finite set of states

Σ is a finite set of input symbols / alphabets

q_0 is Initial state

F is Finite set of Finital states $F \subseteq Q$

δ is transition function

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

Note:-

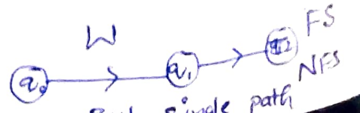
→ By default the FA is DFA

→ DFA has unique Initial state

→ DFA can have any number of final states

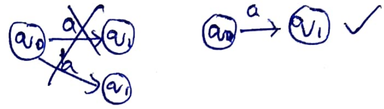
No of FS of DFA is 0 or 1 or more ≥ 0

→ The FA should be Deterministic

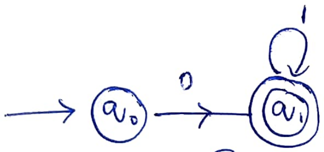


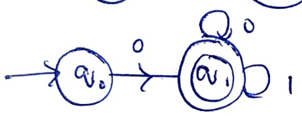
→ DFA is complete system which responses for each and every input symbol. Hence we need to define the transition for each and every symbol from each and every state.

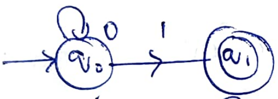
* → In DFA after reading symbol from a state DFA exactly moves to only one state as Next state

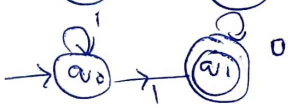
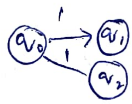


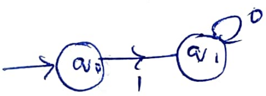
Ex:

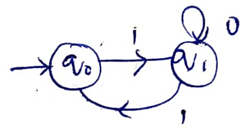
(1)  ✓ DFA

(2)  ✓ DFA

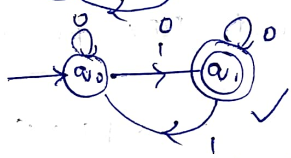
(3)  ✓ DFA

(4)  ✗ 

(5)  ✓ DFA

(6)  ✓ DFA

(7)  ✓ DFA (Initial state can be final state)

(8)  ✓

DFA and also complete DFA

$\boxed{\text{No. of transition} = n(\Sigma) \times n(\theta)}$

→ $\Sigma = \{0,1\} \Rightarrow \Sigma^*$ is Univeral language defined over Σ

$\Sigma^* = \{\epsilon, 0, 1, 01, 10, 00, 11, \ldots\} = \bigcup_{i=0}^{\infty} \Sigma^i$

$\Rightarrow \Sigma^* = (0+1)^*$

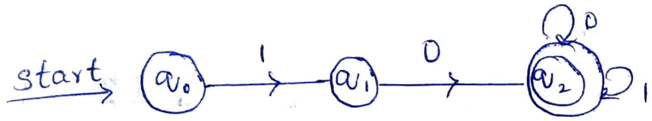
$\Sigma = \{0\} \Rightarrow \Sigma^* = \{\epsilon, 0, 00, 000, 0000, \ldots\ldots\} = 0^*$

$\Sigma = \{1\} \Rightarrow \Sigma^* = \{\epsilon, 1, 11, 111, 1111, \ldots\ldots\} = 1^*$

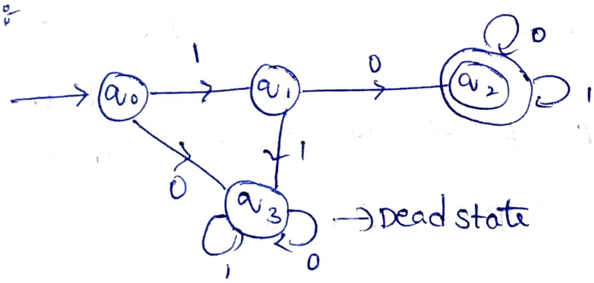
$\Sigma = \{0,1\} \Rightarrow \Sigma^* = \{\epsilon, 0, 1, 00, 10, 01, 11, \ldots\ldots\} = (0+1)^*$

$(01)^* \Rightarrow \Sigma^* = \{\epsilon, 01, 0101, 010101, \ldots\}$

FA :-



CDFA :-

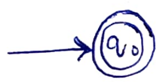


Note :-

① If FA accepts the empty string "ϵ" $\Leftrightarrow$ Initial state is final state

Ex:- $L = \{\epsilon\}$

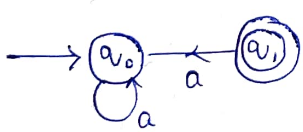
FA :-



$\delta(q_0, \epsilon) = q_0$

② Unreachable state:- The state which can't be reached from the initial state is Unreachable state.

Ex:-

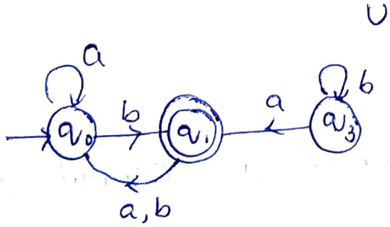


$L = \{\emptyset\}$

No strings accepted

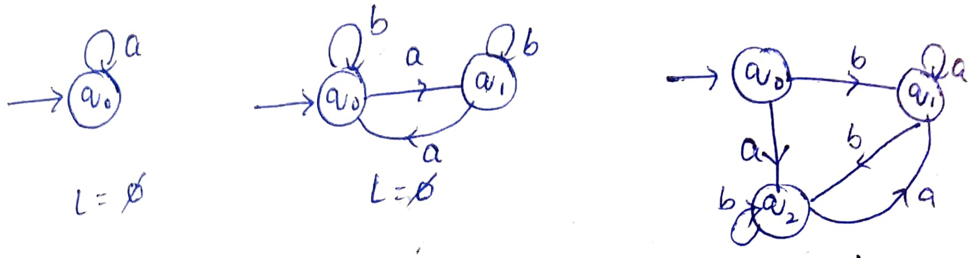
US = q_1

Ex:-



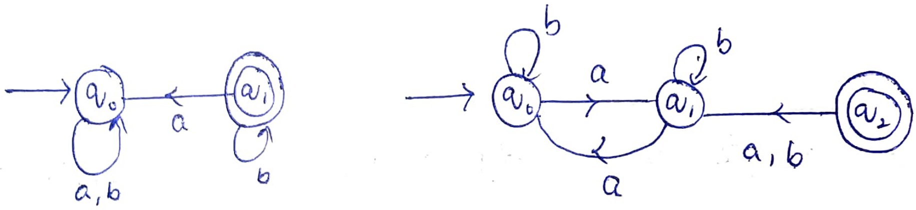
$L = \{ aa\ldots\ldots b,\ aa$

③ FA without Final state accepts <u>Empty</u> language



$L = \emptyset$ $\quad$ $L = \emptyset$ $\quad$ $L = \emptyset$

④ FA in which all the final states are unreachable, such FA accepts <u>Empty</u> language

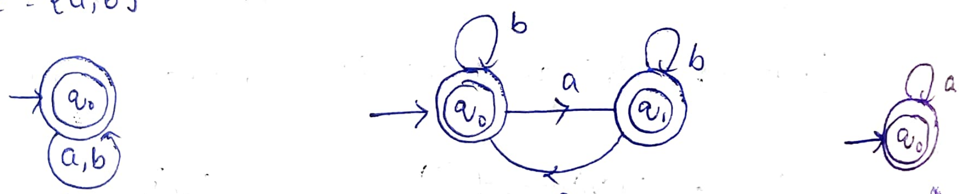


$L = \emptyset$ $\quad$ $L = \emptyset$

⑤ The FA in which All the states are final states Accept the Universal language.

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

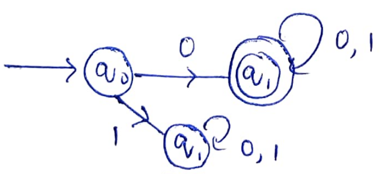
ex:-



$(a+b)^* = \Sigma^*$ $\quad$ $(a+b)^* \Rightarrow \epsilon^*$ $\quad$ a^*

<u>Dead state :-</u>

The states from which we cannot come back is called dead state



Deadstate

Note :- A FA can have atmost one deadstate *problems*

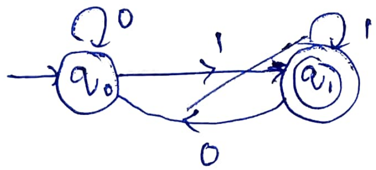
① construct DFA for the language defined over Σ where $L = \{ W \in (0+1)^* \,/\, w$ ~~have~~ ~~even~~ ~~number~~ ~~0~~ starts with 0 and ends with 1 $\}$

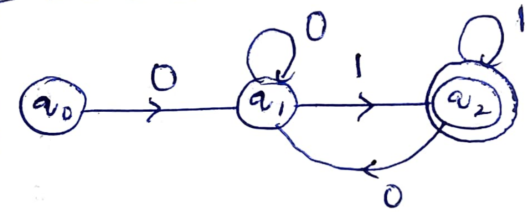
$L = \{ 01, 001, 011, 0001, 0011, 0101, 0111, \ldots \}$

$RL = 0(0+1)^*1$

FA

(Not DFA)
because it is a non-deterministic state





① construct DFA for the language defined over Σ

① $L = \{ W \in (a+b)^* \,/$ every string starts with $ab \}$

② $L = \{ W \in (a+b)^* \,/$ every string strats ~~with~~ and ends with $a \}$

③ $L = \{ W \in (a+b)^* \,/\, |W| = 2 \}$

④ $L = \{ W \in (a+b)^* \,/\, |W| \geq 2 \}$

⑤ $L = \{ W \in (a+b)^* \,/\, |W| \leq 2 \}$

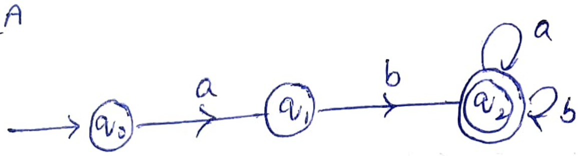
① Sol:- $L = \{ab, aba, abb, abaa, \ldots\}$

abab

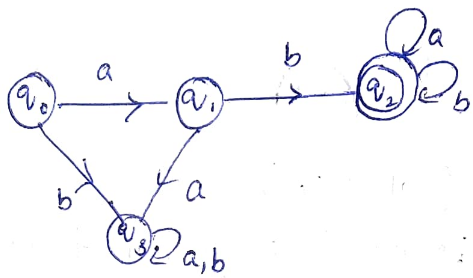
abba

abbb

DFA



Complete DFA



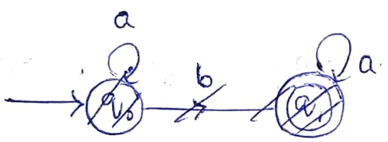
② Sol:- $L = \{a, aa, aba, aaaa, \ldots\}$

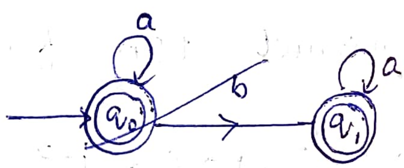
aaa

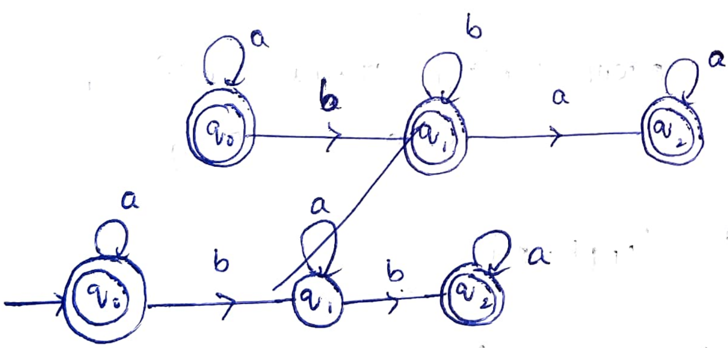
aaba

abaa

abba





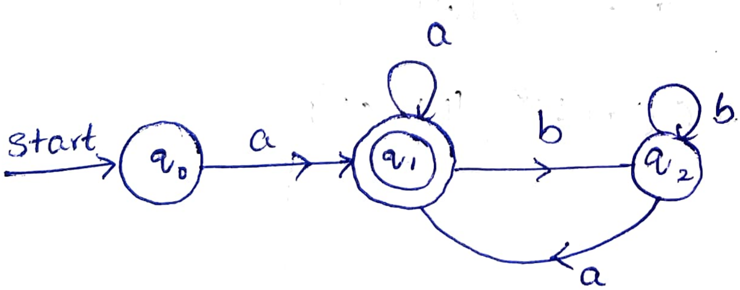


ababa

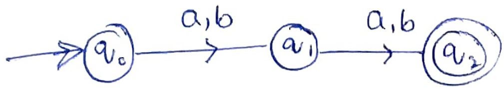
abbaa

aabba

DFA

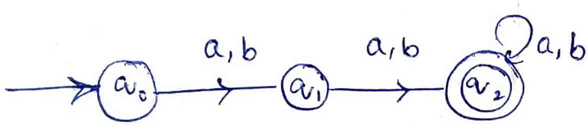


③ sol:- $L = \{aa, ab, ba, bb\}$

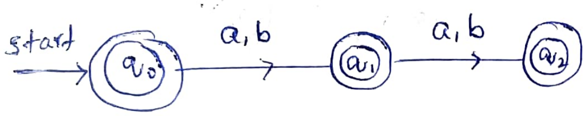


④ sol:- $L = \{aa, ab, ba, bb, aaa, \quad aaaa\}$

aab
aba
baa
⋮
bbb \quad bbbb



⑤ sol:- $L = \{\epsilon, a, b, aa, ab, ba, bb\}$



→ Design DFA for the language defined over $\Sigma = \{a, b\}$

① $L = \{W \in (a+b)^* \,/\, W$ ends with ba$\}$

② $L = \{W \in (a+b)^* \,/\, W$ ends with baa$\}$

③ $L = \{W \in (a+b)^* \,/\, W$ ends with aaa$\}$

④ $L = \{W \in (a+b)^* \,/\, W$ starts and ends with same symbol

⑤ $L = \{W \in (a+b)^* \,/\, W$ starts and ends with different symbol$\}$

⑥ $L = \{W \in (a+b)^* \,/\, W$ has an occurance of ab$\}$

⑦ $L = \{W \in (a+b)^* \,/\, |W|$ is divisible by 2$\}$

⑧ $L = \{W \in (0+1)^* \,/\, |W|$ is divisible by 3$\}$.

⑨ $L = \{W \in (0+1)^* \,/\, W$ has the substring 010$\}$

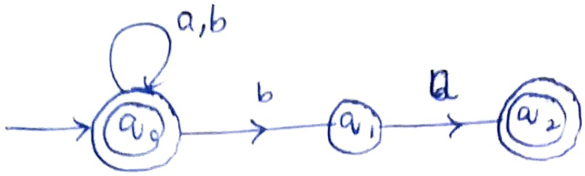
⑩ $L = \{W \in (0+1+2)^* \,/\, |W|$ divisible by 3$\}$

① sol:- $L = \{$ ba, aba, bba, abba, aaaba, ... $\}$

aaba

baba

bbba



② sol:- $L = \{$ baa, abaa, abbaa, ... $\}$

bbaa

babaa

bbbaa

aabaa

aaabba

ababaa

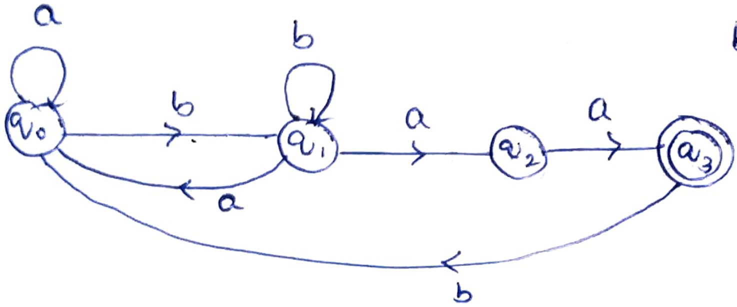
bbbbaa

babbabaa

ababbaa

aaaaa

a



① sol:- $L = \{$ ba, aba, aaba, aaaba, aaaaba $\}$

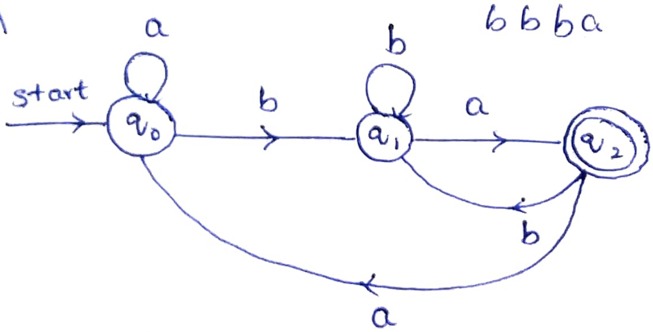
bba, abba, aabba, ababba

baba, ababa

bbba, baaba

⋮

DFA

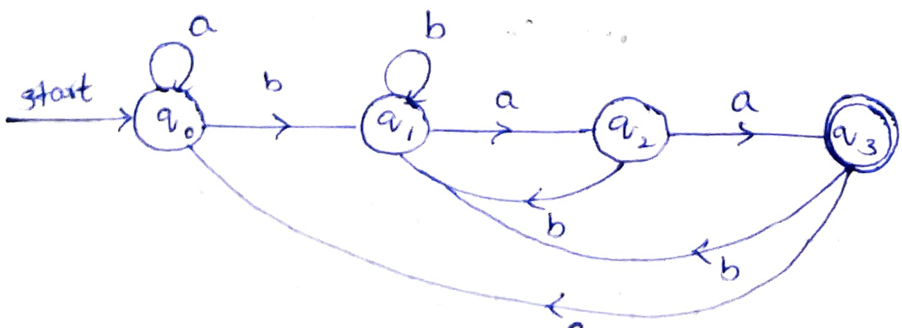


② sol:- $L = \{$ baa, abaa, abbaa, ... $\}$

babaa

bbbaa

DFA



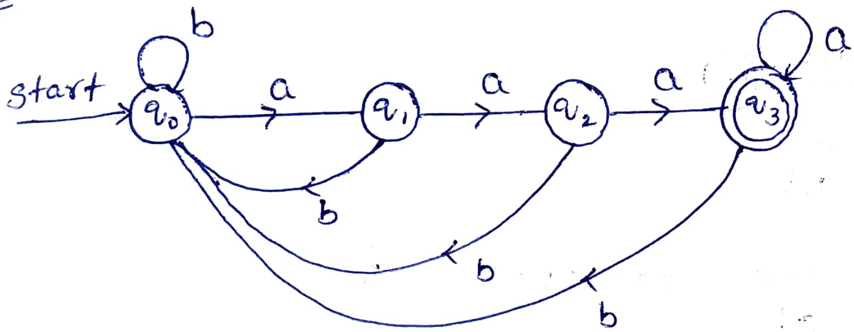
③ sol:- $L = \{$ aaa, abaaa, abaaaa, ... $\}$

aaaaa, aabaaa

bbaaa

baaaa

DFA



④ sol:- $L = \{$ a, aa, aba, abaa, ... $\}$

b, bb, aaa, aaaa

bab, abba

bbb, aaba

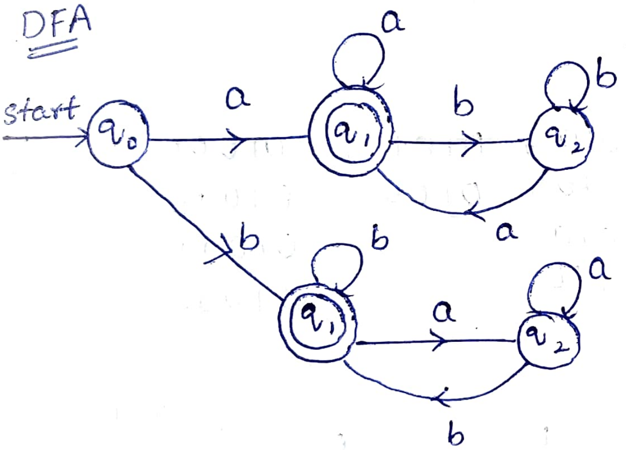
bbab

baab

bbbb

babb

DFA



⑤ sol:- $L = \{$ ab, aab, abab, ... $\}$

ba, abb, aabb

baa, aaab

bba, abbb

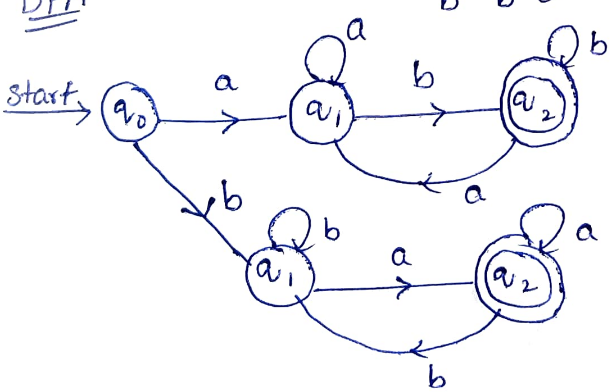
bbaa

baba

baaa

bbba

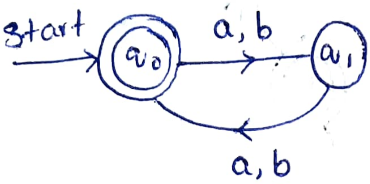
DFA



⑦ Sol: $L = \{\epsilon, aa, \ldots\}$

ab	aabb	:
ba	aaba	:
bb	bbbb	:
	:	

DFA

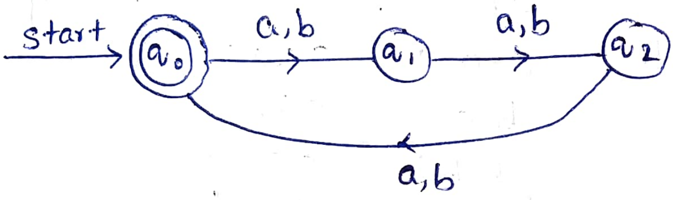


⑧ Sol:

$L = \{\epsilon, aaa, aaaaaaa, \ldots\}$

aba	:
bba	:
:	

DFA

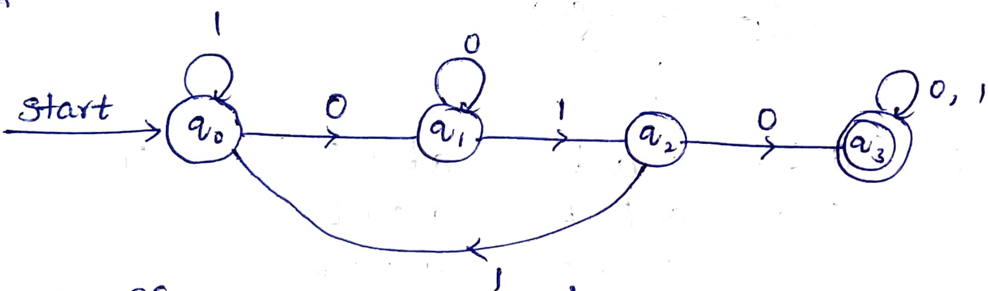


⑨ Sol:

$L = \{010, 1010, 01010, 0101, 01001, \ldots\}$

0010	11010	0100	01010
	10010		01011
	00010		01000

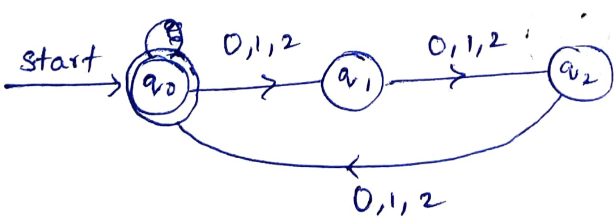
DFA



⑩ Sol: $L = \{\epsilon, 012, 012000, \ldots\}$

120	:
000	:
220	:
:	

DFA



(5) sol:

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

bab   abab   abb

bbab

baab

DFA