UNIT-I

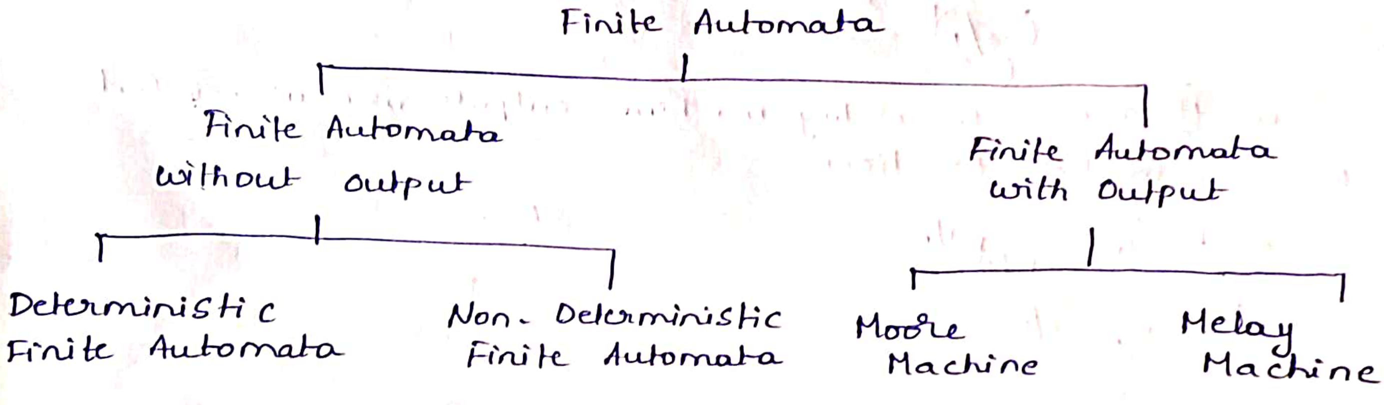
1. List and explain classifications of finite Automata. Discuss the applications of it.

Finite Automata is classified into two types:

1. Finite Automata without output
2. Finite Automata with output.

* Finite Automata without output is classified in to, two types.
  1. Deterministic Finite Automata [DFA]
  2. Non-Deterministic Finite Automata [NFA].

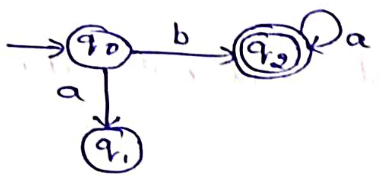
* Finite Automata with output is classified into two types.
  1. Moore Machine
  2. Melay Machine.



1. Deterministic Finite Automata DFA: There is only one path for specific input symbol from current state to next state.

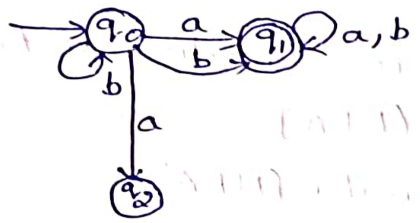
It does not allows 'ε' transitions

Ex:



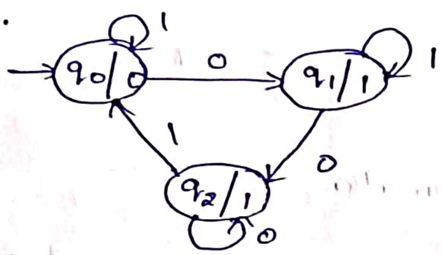
Non-Deterministic Finite Automata: It exists many paths for specific input symbol from current state to next state. It allows 'ε' transition.

Ex:

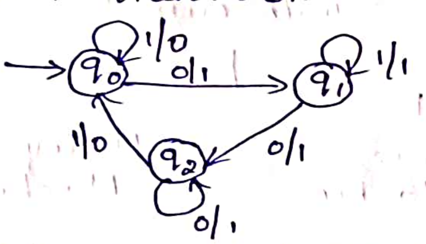


Moore Machine: In moore machine outputs are associated with states.

Ex:



Melay Machine: In melay machine outputs are associated with transition function.



Applications of Finite Automata:

* Finite Automata is used to design lexical Analysis in Compiler design.
* FA is used to create text editor.
* FA is used for spell checking.
* FA is used to design sequential circuits.

2. List the various operations on languages in detail and relate with transition diagrams.

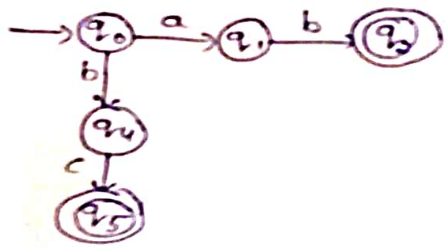
Ans: Operations on Language:

1. Union operation: The Union of two languages L and M, it is denoted with $L \cup M$, is the set of strings that are in both L & M.

Ex: $L = \{ab\}$ and $M = \{bc\} \rightarrow L \cup M = \{abbc\}$

$L = L \cup M \Rightarrow L = L + M$

Transition diagram for $L \cup M$:



2. Concatenation Operation: It combines two string by putting them one after the other. It is denoted with '$\cdot$'

Ex: $X = abc$, $Y = def$ $X \cdot Y = abcdef$

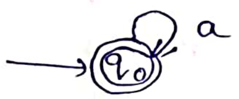
Transition diagram for $X \cdot Y$

3. **Kleene closure**: It performs any no. of combinations. (or) zero or more no. of occurrances. It is denoted with '*'.

Ex: $L = a^*$,

$L = \{\varepsilon, a, aa, aaa \ldots\}$

Transition diagram for a^*



4. **Positive closure**: It performs more than one combinations. It is denoted with '+'.

Ex: $L = a^+$

$L = \{a, aa, aaa \ldots\}$

Transition diagram for a^+.

3. Explain the formal definition of NFA with a suitable example.

Non-Deterministic Finite Automata [NFA]:

→ NFA exists many paths for specific input symbol from current state to next state.

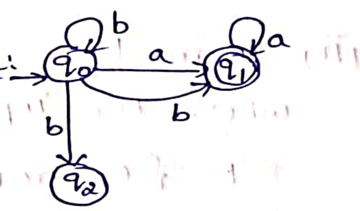
→ NFA is easy to construct than DFA.

→ NFA allows 'ε' transitions

→ Every NFA is not DFA.

→ NFA accepts the empty symbol ε, also contains multiple final states.

NFA contains Five tuples:

Ex: 

M ~~NFA~~ : $\{Q, \Sigma, \delta, q_0, F\}$

Q : Finite set of states.

Σ : Finite set of i/p symbols.

δ : Transition function $Q \times \Sigma \to 2^Q$

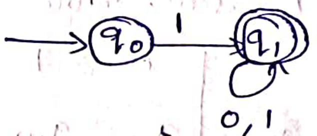
q_0 : Initial state.

F : Final state.

Ex: Design NFA for the input symbols {0,1} and the string starts with 10 or 11

Sol: $\Sigma = \{10, 11, 101, 11011, 10001, 1100011 \ldots\ldots\}$

Transition diagram :



Mapping flow: ~~δ(...)~~

$\delta(q_0, 11011)$

$\delta(q_1, 1011)$

$\delta(q_1, 011)$

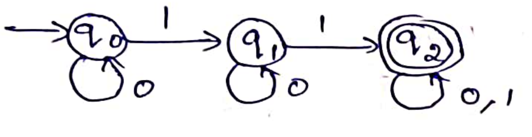
$\delta(q_1, 11)$

$\delta(q_1, 1)$

q_1

4. (i) Draw DFA which accepts the string ending with '11' where the input is {0,1}.

$L = \{11, 0111, 001011, 011, 11011, 1011, 0011, 1111 \ldots\}$



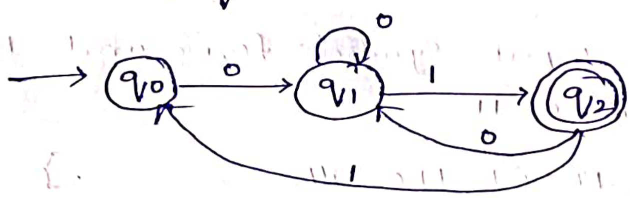
Q/Σ	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_1	q_2
q_2	q_2	q_2

(ii) Draw DFA which accepts the string ending with '01' where the input is {0,1}.

$L = \{01, 001, 101, 0001, 11001, \ldots\}$

$\Sigma = \{a, b\}$ $\quad \Sigma = \{0,1\}$

Transition diagram:-



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

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

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

$q_0 = q_0$

$F = \{q_2\}$

Transition function:-

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

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

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

Transition Table:-

	0	1
q_0	q_1	q_0
q_1	q_1	q_2
q_2	q_1	q_0.

5. Demonstrate the Mathematical definition of DFA. Design DFA which accepts even no of a's and even no of b's. Where the input is a, b.

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

Q = finite set of states

Σ = finite set of input states/symbols

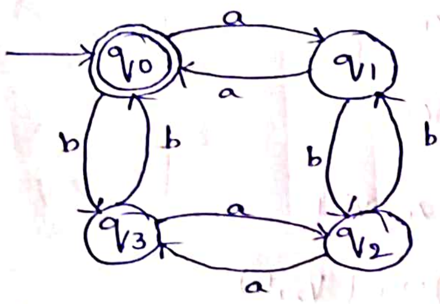
δ = Transition function $Q \times \Sigma \rightarrow Q$.

q_0 = Initial state

F = Final state.

i. Given no of a's and even no of b's.

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



Transition function:-

$\delta(q_0, a) = q_1$ $\quad \delta(q_2, a) = q_3$

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

$\delta(q_1, a) = q_0$ $\quad \delta(q_3, a) = q_2$

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

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

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

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

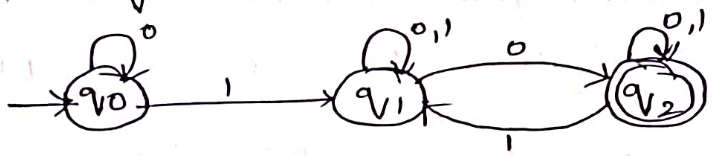
$q_0 = q_0$

$F = \{q_0\}$

Transition Table:-

δ	a	b
q_0	q_1	q_2
q_1	q_0	q_2
q_2	q_3	q_0
q_3	q_2	q_0

7, Convert the given NFA to equivalent DFA.



Ans:-

Step:-1, construct a transition table for given NFA.

	0	1
q_0	q_0	q_1
q_1	$\{q_1, q_2\}$	q_1
q_2	q_2	$\{q_2, q_1\}$

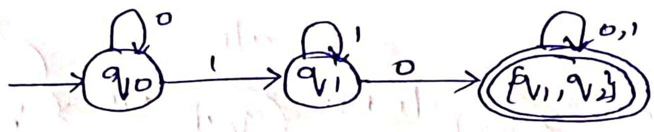
Step:-2 Construct the transition table for DFA.

	0	1
q_0	q_0	q_1
q_1	$\{q_1, q_2\}$	q_1
$\{q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_1, q_2\}$

$$\delta((q_1, q_2), 0) = \delta(q_1, 0) \cup \delta(q_2, 0)$$
$$= \{q_1, q_2\} \cup q_2$$
$$= \{q_1 q_2\}$$

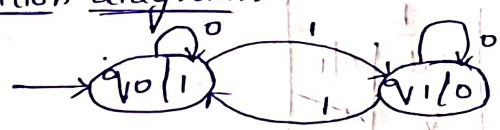
$$\delta((q_1, q_2), 1) = \delta(q_1, 1) \cup \delta(q_2, 1)$$
$$= \{q_1, q_2\} \cup \{q_2, q_1\}$$
$$= \{q_1, q_2\}$$

3, Draw the transition diagram by using DFA transition table



8, Construct a Moore Machine that determines whether an input string contains an even or odd number of 1's. The Machine should give 1 as output if an even number of 1's are in the string and 0 otherwise.

Transition diagram:-



$Q = \{q_0, q_1\}$

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

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

$q_0 = \{q_0\}$

$Q \times \Sigma \rightarrow Q$.

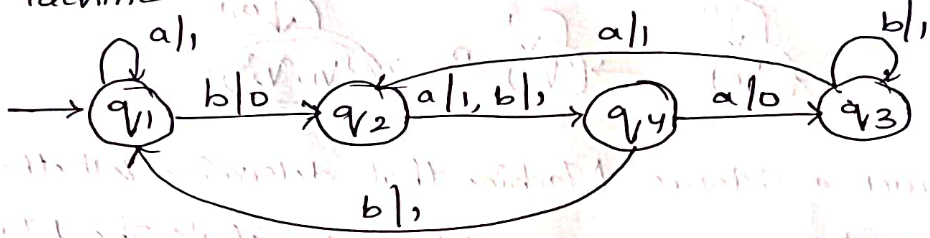
transition table:-

	0	1	o/p
q_0	q_0	q_1	1
q_1	q_1	q_0	0

let us consider the i/p string $1111 \rightarrow$

$$\begin{array}{ccccccccc} \rightarrow q_0 & \xrightarrow{1} & q_1 & \xrightarrow{1} & q_0 & \xrightarrow{1} & q_1 & \xrightarrow{1} & q_0 \\ \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\ 1 & & 0 & & 1 & & 0 & & 1 \end{array}$$

10, Convert the following mealy Machine into equivalent Moore Machine



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

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

Step:-1) Construct a Transition table for given Mealy Machine

Present state	a		b	
	state	o/p	state	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_1	1

~~Step2:- Construct transition table for moore machine~~

Current State	next state a	b	o/p

<u>Step 2:</u> Conversion of Melay to Moore.

$\delta'([q_1, 0], a] = [\delta(q_1, a), \lambda(q_1, a)]$
$= [q_1, 1]$

$\delta'([q_1, 0], b] = [\delta(q_1, b), \lambda(q_1, b)]$
$= [q_2, 0]$

$\delta'([q_1, 1), a) = [\delta(q_1, a), \lambda(q_1, a)]$
$= [q_1, 1]$

$\delta'([q_1, 0], b) = [\delta(q_1, b), \lambda(q_1, b)]$
$= [q_2, 0]$

$\delta'([q_2, 0], a) = [\delta[q_2, a), \lambda(q_2, a)]$
$= [q_4, 1]$

$\delta'([q_2, 0], b) = [\delta(q_2, b), \lambda(q_2, b)]$
$= [q_4, 1]$

$\delta'([q_2, 1], a) = [\delta(q_2, a), \lambda(q_2, a)]$
$= [q_4, 1]$

$\delta'([q_2, 1], b) = [\delta(q_2, b), \lambda(q_2, b)]$
$= [q_4, 1]$

$\delta'([q_3, 0], a) = [\delta(q_3, a), \lambda(q_3, a)]$
$= [q_2, 1]$

$\delta'([q_3, 0], b) = [\delta(q_3, b), \lambda(q_3, b)]$
$= [q_3, 1]$

$\delta'([q_3, 1), a) = [\delta(q_3, a), \lambda(q_3, b)]$
$= [q_2, 1]$

$\delta'([q_3, 1), b) = [\delta(q_3, b), \lambda(q_3, b)] = [q_3, 1]$

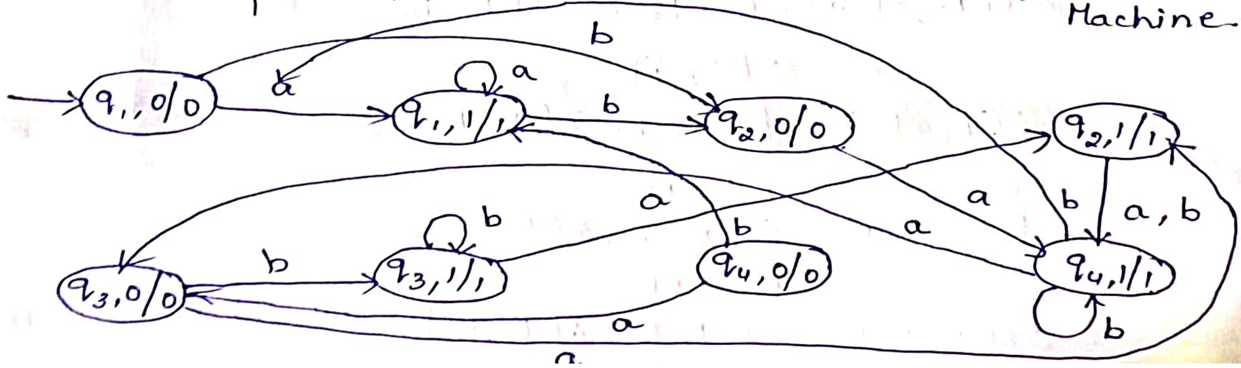
$\delta'([q_4, 0], a) = [\delta(q_4, a), \lambda(q_4, a)]$
$= [q_3, 0]$
$\delta'([q_4, 0], b] = [\delta(q_4, b), \lambda(q_4, b)]$
$= [q_1, 1]$
$\delta'([q_4, 1], a] = [\delta(q_4, a), \lambda(q_4, a)] = [q_3, 0]$
$\delta'([q_4, 1], b] = [\delta(q_4, b), \lambda(q_4, b)] = [q_1, 1]$.

$\lambda'[q_1, 0] = 0$ $\lambda'[q_2, 0] = 0$ $\lambda'[q_3, 0] = 0$ $\lambda'[q_4, 0] = 0$
$\lambda'[q_1, 1] = 1$ $\lambda'[q_2, 1] = 1$ $\lambda'[q_3, 1] = 1$ $\lambda'[q_4, 1] = 1$

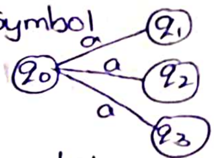
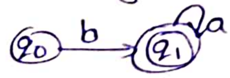
Step 3: Transition Table for Moore Machine

$Q \backslash \Sigma$	a	b	O/P
→ $[q_1, 0]$	$[q_1, 1]$	$[q_2, 0]$	0
$[q_1, 1]$	$[q_1, 1]$	$[q_2, 0]$	1
$[q_2, 0]$	$[q_4, 1]$	$[q_4, 1]$	0
$[q_2, 1]$	$[q_4, 1]$	$[q_4, 1]$	1
$[q_3, 0]$	$[q_2, 1]$	$[q_3, 1]$	0
$[q_3, 1]$	$[q_2, 1]$	$[q_3, 1]$	1
$[q_4, 0]$	$[q_3, 0]$	$[q_1, 1]$	0
$[q_4, 1]$	$[q_3, 0]$	$[q_1, 1]$	1

Step 4: Transition diagram for Melay Machine



9) Compare and contrast the features of NFA with DFA. what is the importance of ϵ-transitions.

DFA	NFA
* It is defined by using five tuples $M = \{Q, \Sigma, \delta, q_0, F\}$ $\delta = Q \times \Sigma \rightarrow Q$	* It is defined by using five tuples $M = \{Q, \Sigma, \delta, q_0, F\}$ $\delta = Q \times \Sigma \rightarrow 2^Q$
* In case of DFA every transaction generates single State with a particular input symbol ex:- $q_0 \xrightarrow{a} q_1$	* In case of NFA every transaction generates more than one state with a particular input symbol ex:- 
* Every state must use every input symbol in a given alphabet. 	* no need to use every input symbol in a given alphabet. 
* It is very difficult to an construct	* It is an easy to an construct.
* Epilson (ϵ) transition use not allowed.	* Epsilon (ϵ) transition use allowed.
* Backtracking is required.	* Backtracking is not requried.
* It requires more space	* It requires less space.
* Practical implementation of DFA is feasible	* Not feasible Need to an Construct NFA to DFA.

Epsilon Transition :- Epsilon transitions, also known as an λ-transitions or null transitions, are transitions in finite automata (including NFA and ε-NFA) that allow the automaton to move from one state to another without consuming any input symbol.

6. Explain the Procedure for constructing minimum state DFA with an example.

Procedure for constructing minimization of states in DFA.

equivalance states; The two states q_2 and q_2 are equivalent if both states $\delta(q_1, a)$ & $\delta(q_2, a)$ are final / non-final states while minimising. First we have have to find which two states are equal and then represent those two states are one as one respective equivalence.

Algorithm:-

1. We create a set $\Pi_0 = \{\{Q_i^0\}\{Q_2^0\}\}$ where

   $\{Q_i\}$ is the set of final states

   $\{Q_i\}$ is the set of non-final states.

   ∴ This is called 0-equivalence.

2. Now, we construct Π_{k+1} from Π_k

   Let Q_i^k be any subset of Π_k.

   If q_1 and q_2 are two states in Q_i^k

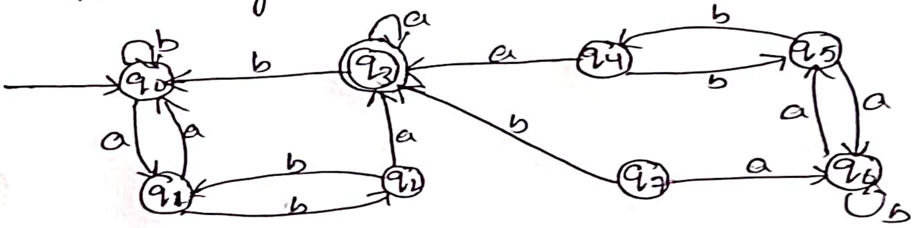
   Have to find the sreliding states in the same equivalence class in Π_k. Then it is said that q_1 + q_2 are k+1 equivalance then Q_i^k is divided into 'k+1' equivalence classes.

3. Repeat step 2 for every Q_i^k in Π_k and obtain all elements in Π_{k+1}

4. continue the above process until $\Pi_n = \Pi_{n+1}$ where $n = 1, 2 \ldots$
5. Then replace all equivalance states in one equivalence class which represents the states

The above process is help in minimizing.

Ex:- construct the state minimizing the FA for the following 'TD'.



Sol:- The given FA is like.

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

where $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7\}$

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

δ : Transition function

q_0 : $\{q_0\}$

F : $\{q_3\}$

$Q \setminus \Sigma$	a	b
q_0	q_1	q_0
q_1	q_0	q_2
q_2	q_3	q_1
q_3	q_3	q_0
q_4	q_3	q_5
q_5	q_6	q_4
q_6	q_5	q_6
q_7	q_6	q_3

0- equivalence.

$\Pi_0 = \{\{q_3\}, \{q_0, q_1, q_2, q_4, q_5, q_6, q_7\}\}$

1- equivalence.

$\Pi_1 = \{\{q_3\}, \{q_0, q_1, q_5, q_6\} \{q_2, q_4, q_7\}\}$

2- equivalence.

$\Pi_2 = \{\{q_3\}, \{q_0, q_6\}, \{q_1, q_5\} \{q_2, q_4\} \{q_7\}\}$

3- equivalence.

$\Pi_3 = \{\{q_3\}, \{q_0, q_6\}, \{q_1, q_5\}, \{q_2, q_4\} \{q_7\}\}$

$\therefore \Pi_3 = \Pi_2$

The minimized FA are.