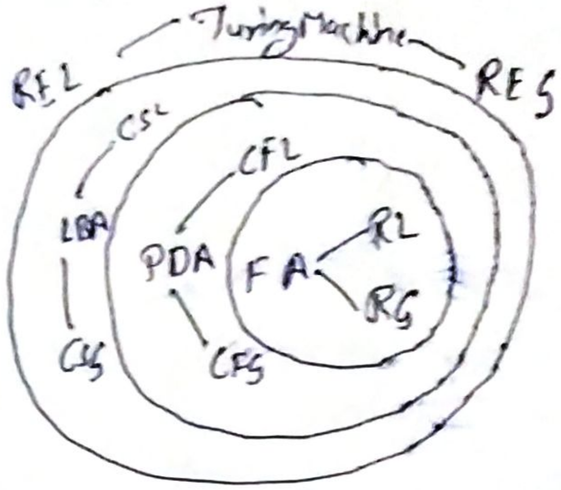
Theory of Automata

Study of Abstract Computing devices.

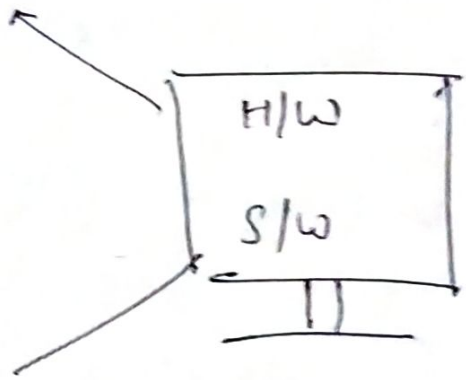
1- Finite Automata (1943)
by McCulloch & Walterpitts.



2- Pushdown Automata (1967)
by Gensburg & Greibach.

3- Linear Bounded Automata (1960)
by John Myhill.

4- Turing Machine (1936) by A. Turing.



Formal language -

Study of Abstract programming concepts.

1- Regular Language & Regular Grammer.

2- Context Free language & context free Grammer.

3- Context-Sensitive language & Context Sensitive grammer.

4- Recursive Enumerable language & grammer.

by Noam chomsky 1956

Alphabets, string, and language

Set of Symbols — list of symbol from alphabet — Set of String.

Symbol — 0 -- 9, a -- z, A -- Z, +, -, %, *, /

Symbol → Alphabet → String → language

Alphabet:- An alphabet is a finite, non empty set of symbol it is denoted by Σ

String:- A string is finite sequence of symbol chosen from an Alphabet

01101 is string from $\Sigma = \{0,1\}$

manish is string from $\Sigma = \{a -- z\}$

(i) Empty String $= \epsilon$

(ii) length of string $= |011| = 3$, $|\epsilon| = 0$

(iii) Power of an alphabet Σ^k

$$\Sigma = \{a, b, c\} = \begin{cases} \Sigma^1 \\ \Sigma^2 \\ \Sigma^3 \end{cases} \text{ Can be computed}$$

$\Sigma^1 = \{a, b, c\}$ $= 3^1$, $\Sigma^2 = \{aa, bb, cc, ab, ac, ba, bc, ca, cb\}$ $= 3^2$

$\Sigma^3 = \{aaa, bbb, ccc, aab, aac, aba, abb, abc \text{ ---- }\} = 3^3$

Σ^* is set of all string so, $\boxed{\Sigma^* = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \text{--} \Sigma^n}$

(iv) Concatenation of string :- Suppose $x = 011$, $y = 101$

$xy = 011101$, $yx = 101011$

$\boxed{\therefore xy \neq yx}$

Language:- A set of strings all of which are chosen from some Σ^* is called language. $\boxed{L \subseteq \Sigma^*}$

language is a subset of Σ^*, $L \subseteq \Sigma^1 \cup \Sigma^2 \cup \Sigma^3$

$\subseteq \{a, b, c, aa, bb, cc, ab, ac, \text{ --- }\}$

$= 39$

Ex:- The language of string consisting of n 0's followed by n 1's for sum $n \geq 0$

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

$\Sigma^* = \{\epsilon, 01, 0011, 000111, 00001111 ---\}$

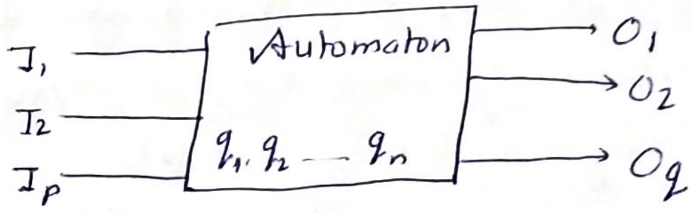
$L = \{\epsilon, 01, 0011, 000111 ....$

Q:- Set of binary number whose value is prime

$L = \{10, 11, 101, 111, 1011 ---\}$

Automata and Grammer

Automata:- An automata is defined as a system where energy, materials and information are transfered, transmitted and used for performing some function without directly particification of human eg. Automatic machine, Automatic packing machine etc.



Charactristics of Automata:-

(i) Input:- Finite number of input from Σ $I_1, I_2 - - I_p$

(ii) output:- Finite number of output $O_1 \cdot O_2 - - O_q$.

(iii) State:- At any instant of time Automaton can be in one of state.
eg. $q_1, q_2 - q_n$

(iv) State Relation → Relation between present input and present state.

(v) Output Relation ;- Either state only or both state and input.

Note: output Depends	input	state	machine Automation	e.g.
	✓	×	without memory	DFA & NFA.
	✓	✓	with memory.	PDA, LBA, TM
	×	✓	moore Machine.	
At any instant of time	✓	✓	Melay Machine.	

Grammer:- The mathematical model of grammer was given by Noam Chomsky in 1956. it turn out to be useful for Computer language BNF (Bakess-Naur-Form) used to describe the definition of grammer.

A grammer is defined as 4-tuple $G = (V, \Sigma, P, S)$

(i) V → finite Non-empty set whose elements are called Non-terminal.

(ii) Σ - finite Non-empty set whose elements are called terminal.

(iii) $V \cap \Sigma = \phi$

(iv) S is special non-terminal ($S \in V$) is start symbol.

(v) P is a set of production rule eg. $\alpha \rightarrow \beta$
where α has at least one Non-terminal.

Note:- (i) Reverse substitution not permitted eg. $S \rightarrow AB$
(ii) No inversion operation permitted if $S \rightarrow AB$
then $AB \rightarrow S$ ×

eg. $G = (\{S, A, B, E\}, \{a, b, c\}, P, S)$

P: $S \rightarrow AB$
$A \rightarrow a$
$B \rightarrow b$
$E \rightarrow c$

BNF Notation   Developed by John Beckes and peter Naur

<symbol> ::= -expression-

<symbol> → Non terminal.

-expression- → Containing one or more symbol ~~sprgh~~
seprated by "/"

(Reduced by) ::= left side must be replace with the expression of Right side.

<Number> ::= <digit>/<digit><number>

<digit> ::= 0/1/2/---9

digit can be replaced by these symbol.

Chomsky hierarchy

Grammer Type	Type 0	Type 1	Type 2	Type 3
1. Grammer Accepted.	Unristricted grammer.	Context Sensitive grammer.	Context free grammer.	Regular Grammer
2. Language Accepted	Recursivelly Enumerable language.	Context Sensitive language.	Context Free. language.	Regular Language
3. Automation.	Turing Machine.	Liner Bounded. Automata.	Pushdown Automata.	Finite state Automata
4. Example	$\alpha \rightarrow \beta$ $Sab \rightarrow ba$ $A \rightarrow S$ $S \rightarrow ACB$	$\alpha A\beta \rightarrow \alpha Y\beta$ $A \in V$ $\alpha \beta Y \in (\Sigma \cup V)^*$ $AB \rightarrow Ab\ Bc$ $A \rightarrow bcA$ $B \rightarrow b$ $aAbcD \rightarrow abcDbcD$	$A \rightarrow Y$ $A \in V$ $Y \in (\Sigma \cup V)^*$ $S \rightarrow Xa$ $X \rightarrow a$ $X \rightarrow aX$ $X \rightarrow \Lambda$	$X \rightarrow a$ or $X \rightarrow aY$ $X\,Y \in V$ $a \in \Sigma$ $S \rightarrow \Lambda$ $X \rightarrow a/aY$ $Y \rightarrow b$ $X \rightarrow \Lambda$

FINITE AUTOMATA

- FA without output
  - DFA
  - NFA
  - ENFA
- FA with output
  - MOORE Machine
  - Meloy Machine

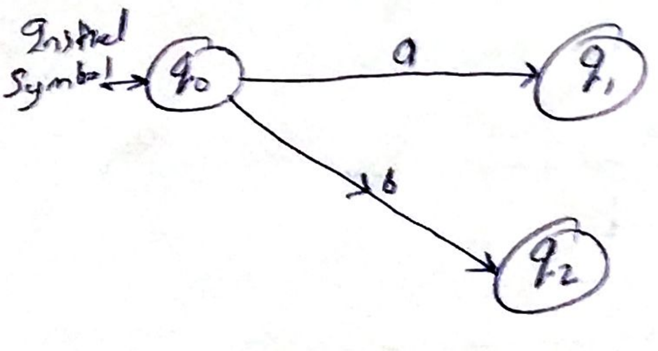
FA - A finite automata has a set of states and its control move from state to state in response to external input

Deterministic Finite Automata (DFA)

In DFA on each input there is only one state which the automata can transition from its current state.

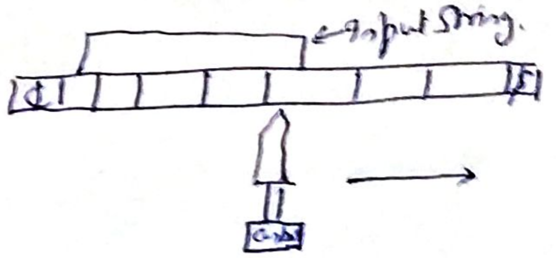
A DFA consist 5-tuple $(Q, \Sigma, \delta, q_0, F)$

$Q \rightarrow$ Finite non-empty set of state
$\Sigma \rightarrow$ Finite non-empty set of input
$\delta \rightarrow$ transition function $Q \times \Sigma \rightarrow Q$
eg. $\delta(q_0, a) \rightarrow q_1$
$q_0 \rightarrow q_0 \in Q$ is Initial state.
$F \subseteq Q$ a set of final state called Acceptance state.



$\delta(q_0, a) \rightarrow q_1$

$Q \times \Sigma$ $Q = \{q_0, q_1, q_2\}$
$\Sigma = \{a, b\}$
$3 \times 2 = 6$



Example:-

(01) should be there for accept

$L = \{01, 0011, 101, 11011 \ldots\}$ a language

So design a machine for this.

DFA for particular language

DFA $A = (\{q_0, q_1, q_2\}\{0,1\}, \delta, q_0, q_2)$

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

Notation of DFA -

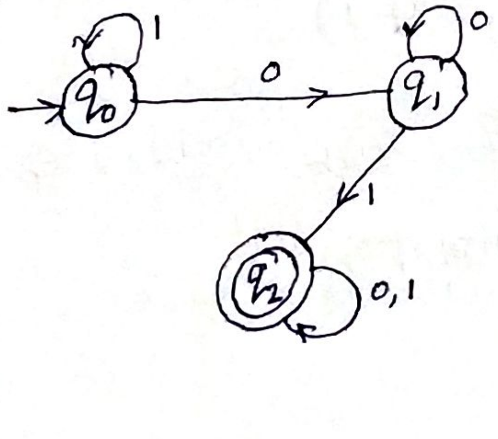
1- Transition Diagram
2- Transition Table.

Example:-

DFA $A = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, q_2)$

$$\delta :- \delta(q_0, 1) \rightarrow q_0$$
$$\delta(q_0, 0) \rightarrow q_1$$
$$\delta(q_1, 0) \rightarrow q_1$$
$$\delta(q_1, 1) \rightarrow q_2$$
$$\delta(q_2, 0) \rightarrow q_2$$
$$\delta(q_2, 1) \rightarrow q_2$$

• Transition Diagram -



Transition Table

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

Properties of Transition function of DFA

$$\delta(q, w) = q$$

No of transition function in DFA = $Q \times \Sigma$ in Q.

Suppose $\{q_0, q_1, q_2\}$ $\{0,1\}$ $\Rightarrow 3 \times 2 = 6$ No of transition function.

ex. $(q_0, 1) \rightarrow q_1$

Properties:- 1- $\delta(q, \lambda) = q$

2- $\boxed{\delta(q, w) = F}$ $w = 1101$

$\delta(q, 1101)$ for this w can be written as.

$w = xa$ $x = 110$ $a = 1$ or $a = 1$ $x = 101$

$w = ax$

So if $w = xa$

$\Rightarrow \delta(q, xa)$

$\Rightarrow \delta(\delta(q, x), a)$

for $w = ax$

$\Rightarrow \delta(q, ax)$

$\Rightarrow \delta(\delta(q, a), x)$

Example:- DFA $A(\{q_0, q_1, q_2, q_3\}, \{0,1\}, \delta, q_0, q_0)$

String $w = 11010$

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

$\delta(q_0, w) = \delta(q_0, 110101)$
$= \delta(q_1, 10101)$
$= \delta(q_0, 0101)$
$= \delta(q_2, 101)$
$= \delta(q_3, 01)$
$= \delta(q_1, 1) = \delta(q_0, \wedge)$
$= ~~\delta q_0~~ \quad = q_0$

So the string 110101 is acceptable by this machine.

(7) Language of DFA - A language for DFA $A = (Q, \Sigma, \delta, q_0, F)$ define as $L(A) = \{w \mid \hat{\delta}(q_0, w) \text{ is in } F\}$

→ w is a string. q_0 is initial state.

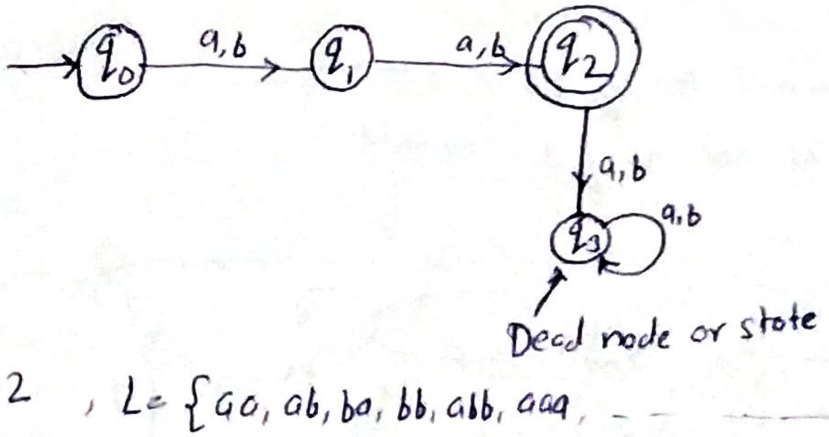
And the language which is acceptable in DFA is Regular language.
DFA basically accept or reject a language.

How ~~t~~ to construct DFA for particular language.

Q1- Construct a DFA that Accept all string over {a, b}
(i) $|w| = 2$ (ii) $|w| \geq 2$ (iii) $|w| \leq 2$

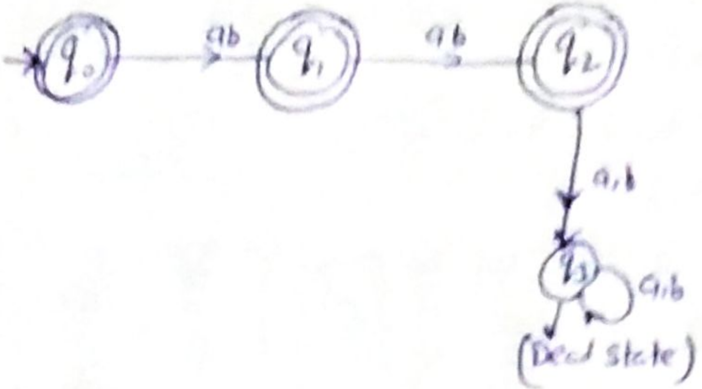
(1) $|w| = 2$

(a, b) L = {aa, bb, ab, ba}



(ii) $|w| \geq 2$, L = {aa, ab, ba, bb, abb, aaa, ------ }

(iii) $|w| \leq 2$ $L = \{\epsilon, a, ab, aa, ba, bb\}$

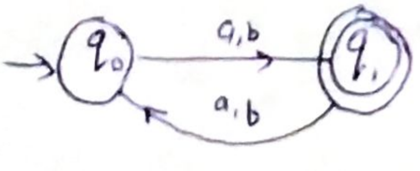


Q2: Construct DFA $w \in \{a,b\}$

(i) $|w| \bmod 2 = 0$ (ii) $|w| \bmod 2 = 1$

(i) $|w| \bmod 2 = 0$, $L = \{\epsilon, ab, ba, aa, bb, aaaa, bbbb, abab \text{ ------- }\}$

(ii) $|w| \bmod 2 = 1$ $L = \{a, b, aaa, bbb, abb, aab, aabbb \text{ ------- }\}$



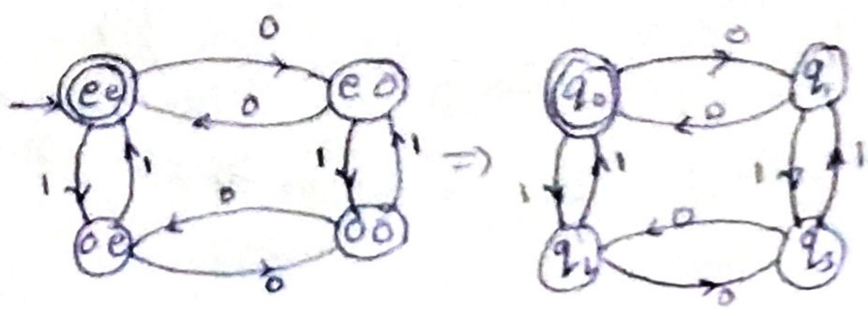
Q3:- Construct DFA accepting following language over $\Sigma = \{0,1\}$

(i) No. of 1's is even and No. of 0's is even.

(ii) No. of 1's is odd and no of 0's is odd.

Ans:- (i)

0	1
even	even
even	odd
odd	even
odd	odd



(i) $L = \{\epsilon,$

(ii)

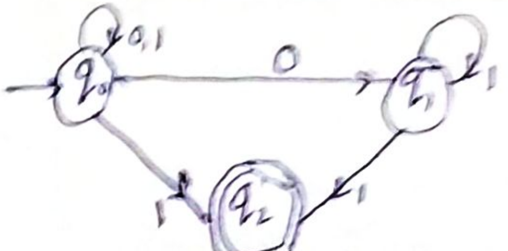
Non-Deterministic Finite Automata
(NDFA/NFA)

A NFA has the power to be in several states at once for NFA 'δ' is a function that takes a state and input symbol as argument but return at zero, one or more state

$\delta(q_0, a) \to q_1 \to$ DFA

$\delta(q_0, a) \to \phi$ or q_1 or $\{q_1, q_2\}$ — NFA

Suppose-



NFA consists 5 tuples $(Q, \Sigma, \delta, q_0, F)$

where $Q \to$ ~~set~~ finite non empty set of states.

$\Sigma \to$ finite non empty set of inputs.

$q_0 \to$ Element of Q $q_0 \in Q$ and is initial state

$F \to F \subseteq Q$ and it is final state

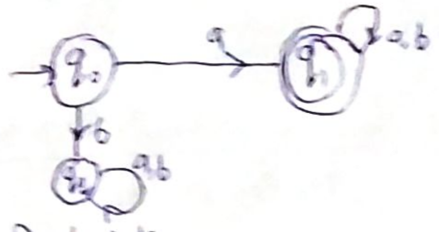
$\delta \to Q \times \Sigma \to 2^Q$

Q:- Construct NFA that accept all string over $\Sigma = \{a, b\}$, where each string Start with 'a'.

Solution:- $L = \{a, aa, ab, abb ---\}$

NFA

DFA



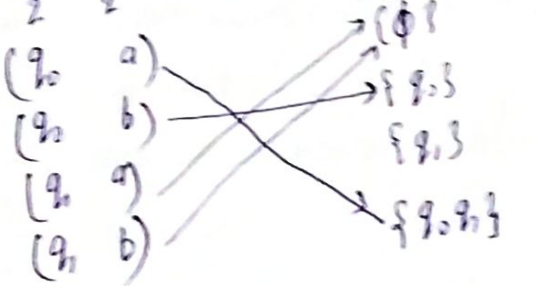
Dead state

Q:- Construct NFA for $L = \{$end with 'a'$\}$ where end with 'a'. that accept all string over $\Sigma = \{a, b\}$

Sol:- $L = \{a, ba, aa, aaa, baa ______\}$

So

$\underset{2}{Q} \times \underset{2}{\Sigma} \rightarrow 2^2 = 4 \Rightarrow 2^Q$

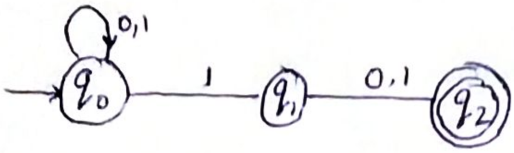


Transition table

	a	b
q_0	$\{q_0 q_1\}$	q_0
q_1	$\{\phi\}$	$\{\phi\}$

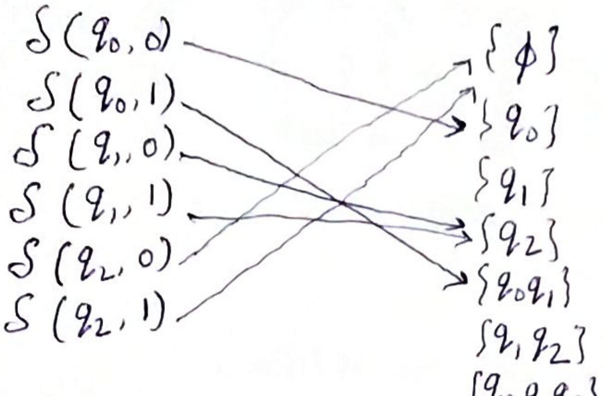
Q 1- Construct NFA accepting the string {0,1}, whose second last symbol is '1'.

Sol:- $L = \{10, 11, 1010, 1110, 0010 \ldots\ldots \}$



Transition table: $Q \times \Sigma \rightarrow 2^Q$

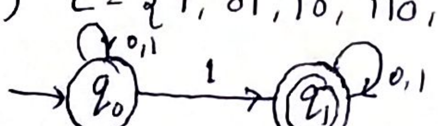
$3 \quad 2 \rightarrow 2^3 = 6$



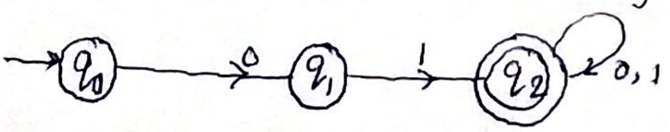
Q2:- Construct the NFA that accept all string over {0,1} where.

(i) String contain '1'
(ii) Start with 01
(iii) Contain 01
(iv) End with 01

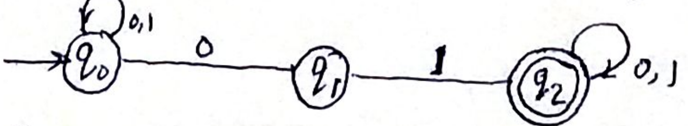
Sol:- (i) $L = \{1, 01, 10, 110, 001 \ldots\ldots \}$



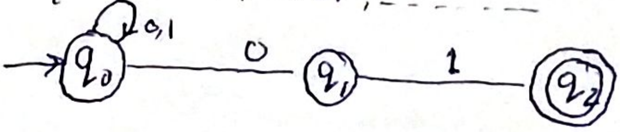
(ii) $L = \{01, 0100, 0111, 0101 \ldots\ldots \}$



(iii) $L = \{01, 0010, 0010100 \ldots\ldots \}$



(iv) $L = \{01, 0001, 1101, \ldots\ldots$



NFA to DFA Conversion –

① Construct DFA equivalent to NFA $m = (\{q_0, q_1\}, \{0,1\}, \delta, q_0, \{q_0\})$

δ

State/Σ	0	1
→ q_0	q_0	q_1
q_1	q_1	$q_0 q_1$

Solution → DFA

1 – $\{q_0, q_1\} = \phi, \{q_0\}, \{q_1\}, \{q_0, q_1\}$ ⇒ Power set

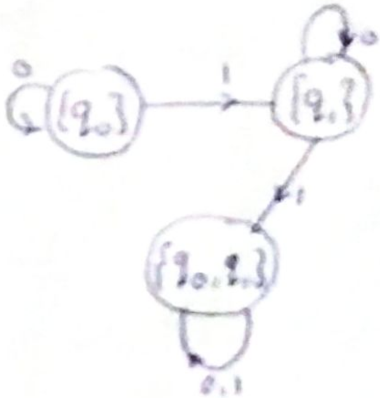
② q_0 in initial [Identify initial state]

③ $\{q_0\}, \{q_0, q_1\}$ [Identify final state]

④ δ table (Transition table)

State/Σ	0	1
ϕ	ϕ	ϕ
→ $\{q_0\}$	$\{q_0\}$	$\{q_1\}$
$\{q_1\}$	$\{q_1\}$	$\{q_0, q_1\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1\}$

⇒



Minimization of DFA

Q: Convert minimum state automata equivalent to finite Automata describe by

State/Σ	0	1
→ q_0	q_1	q_5
q_1	q_0	q_2
q_2	q_0	q_2
q_3	q_2	q_6
q_4	q_2	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2

Sol

π_0 (Zero equivalence) $= \{\{q_2\}, \{q_0, q_1, q_3, q_4, q_5, q_6, q_7\}\}$

π_1 (one equivalence) $= \{\{q_2\}, \{q_0, q_4, q_6\}, \{q_1, q_7\}, \{q_3, q_5\}\}$

$\pi_2 = \{\{q_2\}, \{q_0, q_4\}, \{q_6\}, \{q_1, q_7\}, \{q_3, q_5\}\}$

$\pi_3 = \{\{q_2\}, \{q_0, q_4\}, \{q_6\}, \{q_1, q_7\}, \{q_3, q_5\}\}$

because $\boxed{\pi_2 = \pi_3}$ we will stop here

Now DFA