29/09/21 : NFA & DFA : M ($Q, \Sigma, \delta, q_0, q_f$)

★ Finite Automata : ~~Mill~~ | Mealey Machine & Moore Machine

→ There is output but no final state

★ $\boxed{\text{Moore Machine}}$ (output length one more than input)

↳ It is a Finite Automata ~~with~~ with no ~~output~~ final state & it produces output sequence for given input sequence.

↳ In moore machine, output symbol is associated with each state.

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

Q = Finite no. of states : $\{q_0, q_1, q_2\}$

Σ = Input symbol $\{a, b\}$

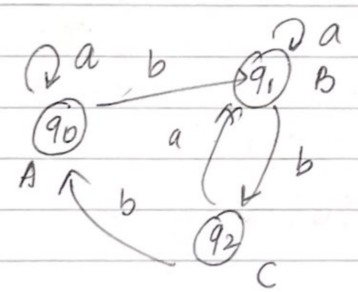
Δ = Output symbol $\{A, B, C\}$

$\delta = Q \times \Sigma \longrightarrow Q$ (transition)

(mapping funcn) $\lambda = Q \longrightarrow \lambda$ (state gives output)

q_0 = Initial state

eg: ↳



moore machine

$\delta : = Q \times \Sigma \longrightarrow Q$

Q/Σ	a	b
q_0	q_0	q_1
q_1	q_1	q_2
q_2	q_1	q_0

Transim table

Mapping funcn : λ :

$q_0 : A$

$q_1 : B$

$q_2 : C$

output symbol associated with state, so print somethin

Input : aba $\boxed{(n)}$

Output AABB $\boxed{(n+1)}$

Initial : $q_0 \xrightarrow{q_0} AA$

Mealy machine (output length equal to input length)

It is a finite automata with no final state & it produces output sequence for the given input symbol.

In mealy machine, output is associated ~~to~~ with each transition.

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

Q = finite no. of states

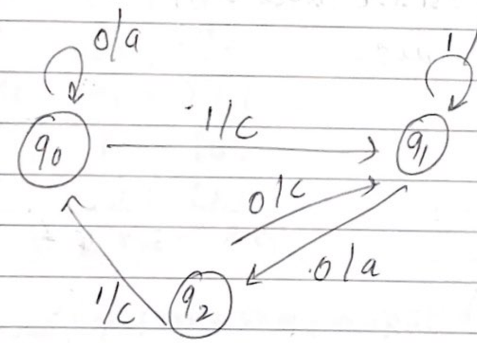
Σ = I/P symbol

Δ = O/P symbol

δ = $Q \times \Sigma \longrightarrow Q$

$\boxed{\lambda = Q \times \Sigma \longrightarrow \Delta}$

q_0 = Initial state



Q\Σ	0	1
q_0	q_0	q_1
q_1	q_2	q_1
q_2	q_1	q_0

$\delta : Q \times \Sigma \longrightarrow Q$

Q\Σ	0	1
q_0	a	c
q_1	a	b
q_2	c	c

$\lambda\Delta : Q \times \Sigma = \Delta$

Input : 010 $\boxed{n}$

output : $\boxed{aca}$ $\boxed{n}$

q_0 on 0 $\longrightarrow q_0$

which is a $\rightarrow q_0$

q_0 on 1 gives $\hookrightarrow q_1$

which is c $\Rightarrow q_0$

q_1 on 0 gives

Q1) Design a Moore machine to output
'A' if string ends in '101'
'B' if string ends in '110'
'C' ~~if string~~ otherwise over $\Sigma \{0,1\}$

VI
$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$

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

$\Delta = \{A, B, C\}$

eg: $L = \{001, 110, 101, 11101 \ldots\}$

Output : C B A .....

I

	Q\Σ	0	1
→	q_s	q_0	q_1
0	q_0	q_0	q_1
1	q_1	q_2	q_4
10	q_2	q_0	q_3
101	q_3	q_2	q_4
11	q_4	q_5	q_4
110	q_5	q_0	q_3

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

II λ

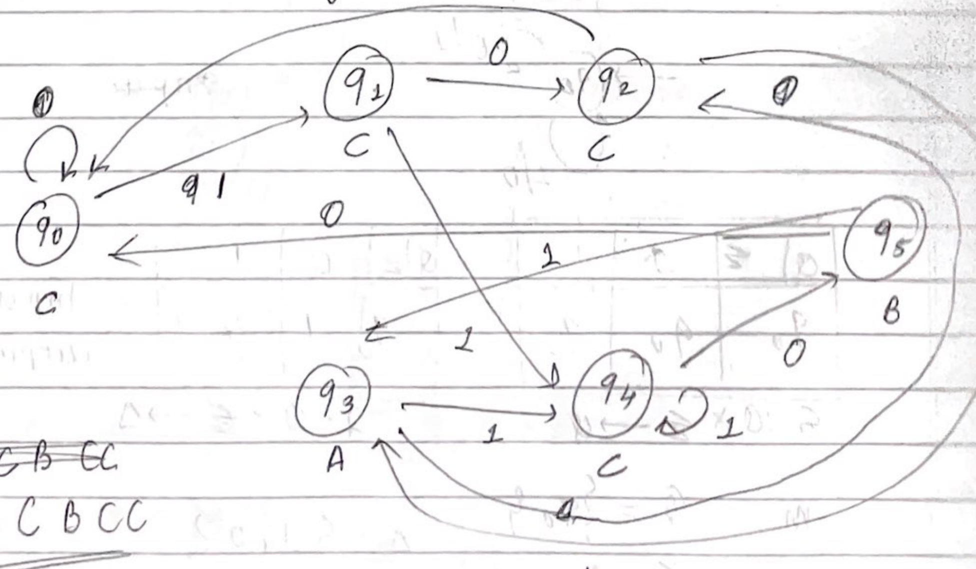
q_s	q_s	C
q_0	q_0	C
q_1	q_1	C
q_2	q_2	C
q_3	q_3	A
q_4	q_4	C
q_5	q_5	B

IV
$\delta(q_s, 101)$
q_1
q_2
q_3 ✓
A

V
Input : 101
output : CCCA

⇒ same input ~~⇒~~ + same output & same symbol ⇒ Then these elements can be merged

III



110 01
~~CCBCBCC~~
CCCBCC

1000 will print 1001

(Q2) Design a moore machine to check accurance of ~~100~~ '1000' to '1001'

A : 1000

B : 1001

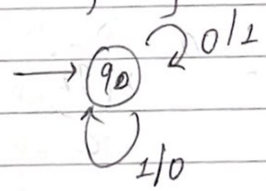
C : otherwise

	Q|Σ	0	1	
	q_s	q_0	q_1	
0	q_0	q_0	q_1	0
1	q_1	q_2	q_1	1
10	q_2	q_3	q_1	0
100	q_3	q_4	q_1	0
1000	q_4	q_0	q_1	1
~~1001~~	~~q_5~~			

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

Mealy machine

(Q3) Design a mealy machine for 1's complement of a number over ~~input~~ ~~{0~~ $\Sigma = \{0, 1\}$



~~input:~~

Q|Σ	0	1
q_0	q_0	q_0

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

Q|Σ	0	1
q_0	1	0

$\lambda : Q \times \Sigma \rightarrow \Delta$

Input : 0110

Output : 1001

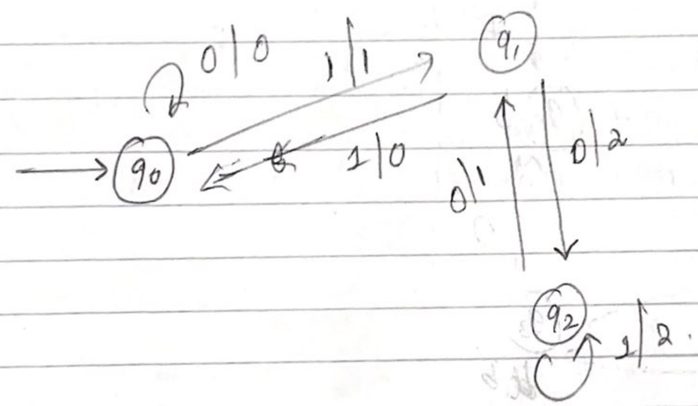
M: $Q = \{q_0\}$ $\quad \Delta = \{1, 0\}$

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

Q4: Design a mealy machine to output remainder when the binary number is divisible by 3

~~Input = {0, 1}~~ {00, 01, 10, ~~11~~0}

Output = {0, 1, 2}



	Q|Σ	0	1
→ 0	q_0	q_0	q_1
0	q_1	q_0	q_1
1	q_2	q_2	q_1
2	q_3	q_2	q_3

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

Q|Σ	0	1
q_0		
q_1		
q_2		
q_3		

$\lambda: Q \times \Sigma \rightarrow \Delta$

	Q|Σ	0	1
0	q_0	q_0	q_1
1	q_1	q_2	q_0
2	q_2	q_1	q_2

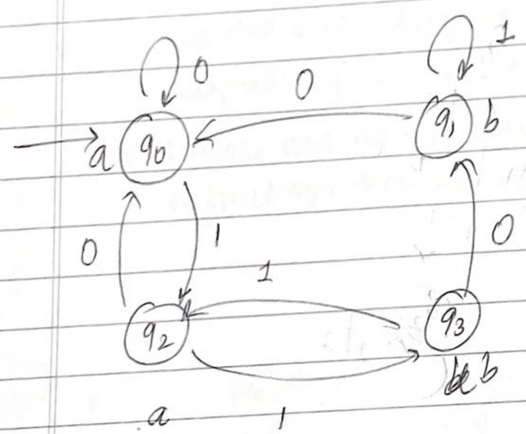
	Q|Σ	0	1
0	q_0	0	1
1	q_1	2	0
2	q_2	1	2

↱ Diagram might be given

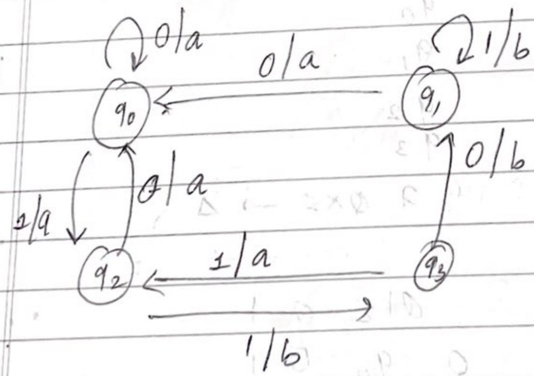
* $\boxed{\text{Moore Machine to Mealy Machine Conversion}}$

↳ Assign the output symbol associated with the state to all its incoming transitions

↳



(A)



☆ **Mealy machine to moore machine conversion**

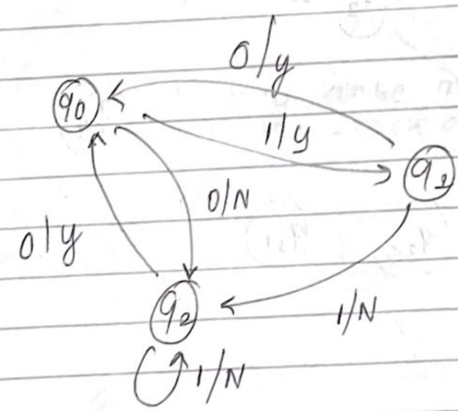
↳ If the output symbol along with the incoming transition to a state are same, then assign that symbol to that state

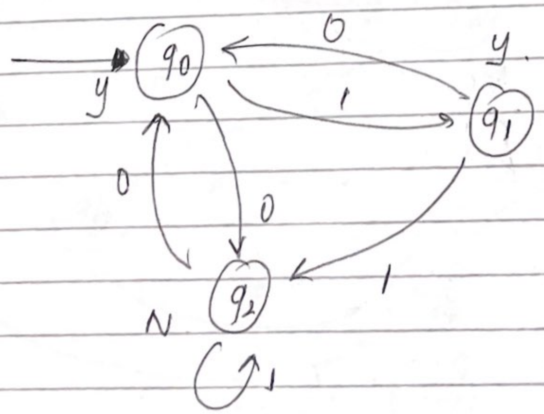
↳ If the output symbol along with incoming transition to a state are not same, then split that state as many times as the output symbols with each state producing a diff output ($q_0 \rightarrow q_{0A}$ & q_{0B})

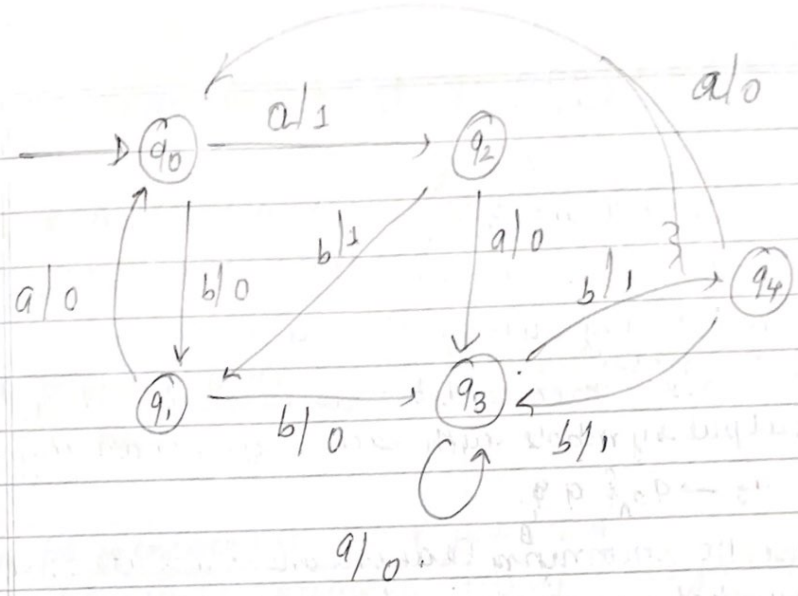
↳ If there are no incoming transitions to a state, then any output symbol can be assigned to that state

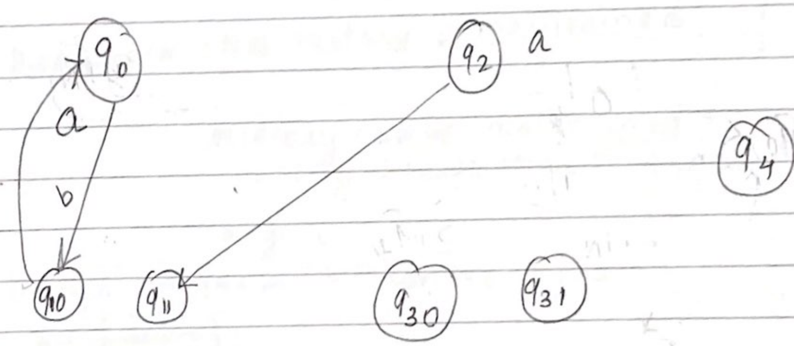
y/N

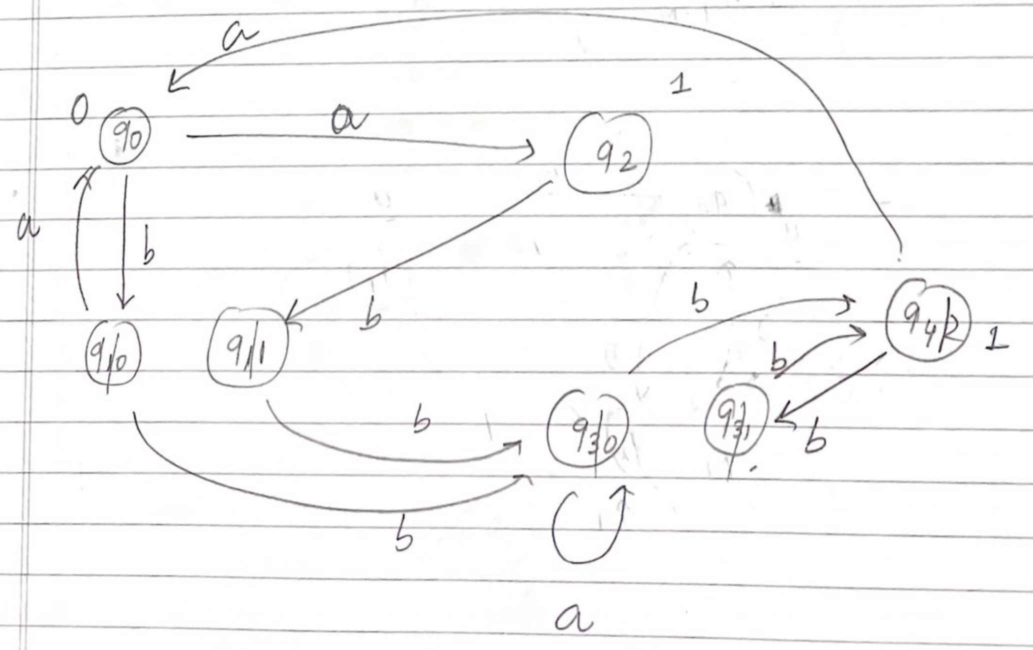
eg (1)






a/1
a/0
b/1
a/0
b/0
a/0
b/1
b/0
b/1
a/0
q0
q1
q2
q3
q4


q0
q2
a
a
b
q4
q10
q11
q30
q31


a
0
q0
a
q2
1
a
b
b
b
b
q4
1
q10
q11
b
q30
q31
b
b
a

↳ Finite Automata : Final state ✓ (NFA & DFA)
↳ Nonfinite automata : ~~mee~~ Final state ×. (mealy & moore)

03/10/22

★ Differentiate ~~bet~~ between DFA & NFA (5)

↳ DFA : can't have ∈ transitions
NFA : can have ∈ transition

} example also include

★ Difference bet. mealy & moore machine / (5)
↳ for n ↳ n. ↳ n+1.

★ Difference bet. FA & ~~pus~~ PDA
FA: Finite Automata ⇒ input ⇒ 1 symbol & move ahead
PDA : Pushed Down Automata
↳ Drawback : No history is maintained

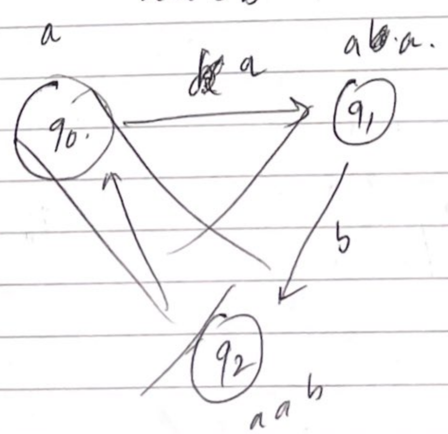
History can be maintained using stack operation (push & pop)

lang L=

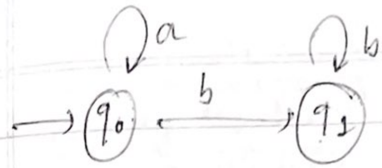
Q1. Design FA for $a^n b^n$ where $n \geq 1$
~~Σ = {a, b}~~

a
~~b~~
ab
~~aab~~
aabb

} ab, a^2b^2, a^3b^3



$\Gamma \Rightarrow$ stack alphabet
tape symbol



$\Longrightarrow$ aaabb

aabb   Can't keep track of the count

So, this is the limitam of FA
Thus, PDA is used which keeps track of the count as well

→ PDA : 7 tuples

z_0: initial stack top state

state change. push pop operation

aabb — push X, pop X
— pop X
— push X

↳ pumping lemma for regular lang