Finite Automata

- without output
  - DFA: $\delta: Q \times \Sigma \to Q$
  - NFA: $\delta: Q \times \Sigma \to 2^Q$
  - ε-NFA: $\delta: Q \times (\Sigma \cup \varepsilon) \to 2^Q$
  - Equivalent in power
- with output
  - Moore
  - Mealy
  - Equivalent in power
  - Deterministic

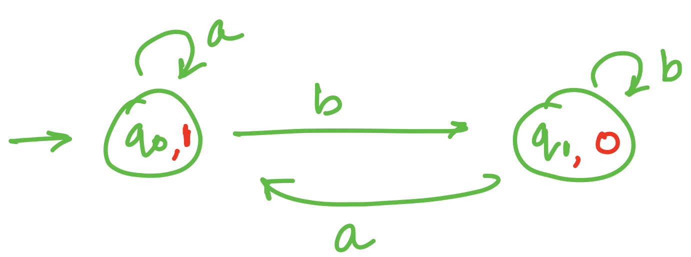
$Q, \Sigma, \delta, q_0, \Delta, \lambda$

- Q → set of states
- Σ → input alphabet
- δ → transition fxn: $Q \times \Sigma \to Q$
- q_0 → initial state
- Δ → output alphabet
- λ → output fxn

FA with output (Deterministic)

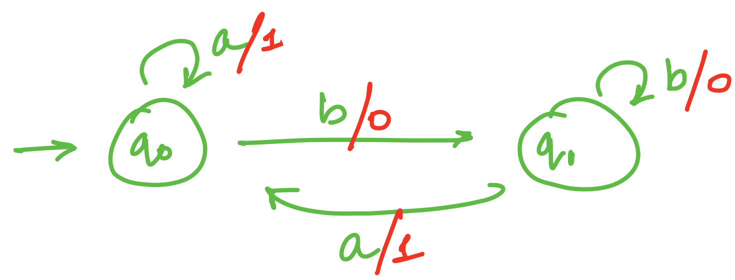
- Moore
- Mealy

Moore



- output is associated with state
- Eg: q_0 : output is 1
  q_1 : output is 0

Mealy



- output is associated with current state & input alphabet
- Eg: $q_0, a \to 1$
  $q_0, b \to 0$
  $q_1, a \to 1$
  $q_1, b \to 0$

Input: ab

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

1 1 0

110

Input: ab

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

1 0

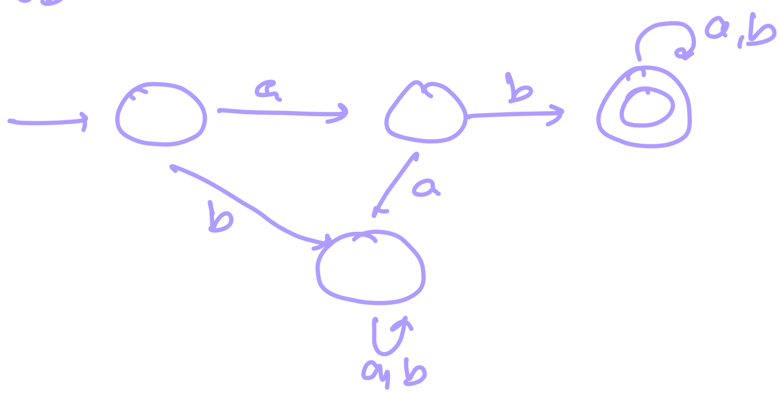
10

Input length: n
Output length: n+1

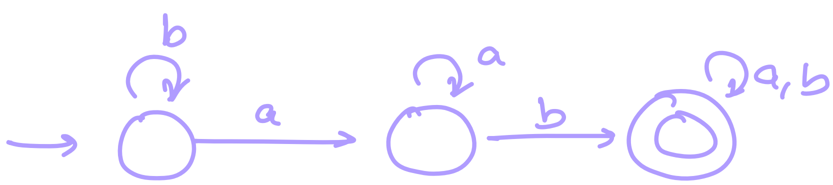
Input length: n
output length: n

Q: Construct a moore m/c that takes set of all strings over {a,b} as input & prints 1's as o/p for every occurrence of 'ab' as a substring.

Starting with ab

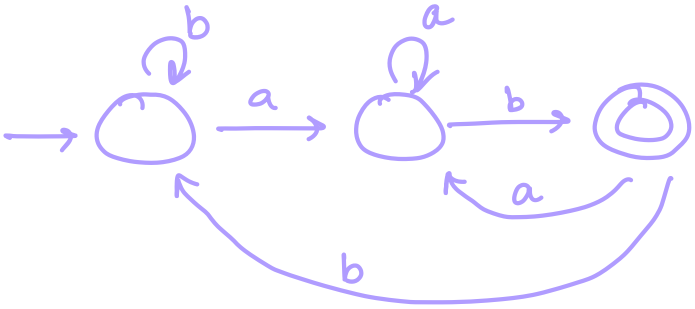


Substring ab

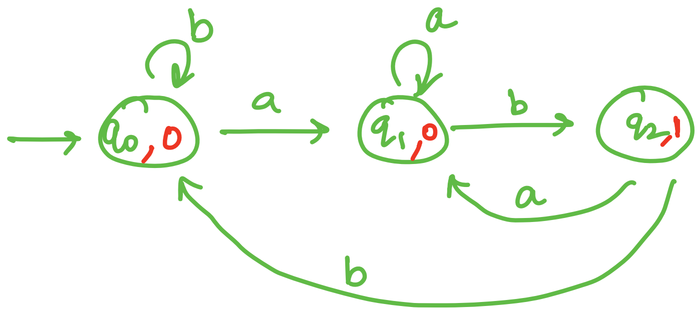


abaab

Ends with ab



ab babaa
1 1



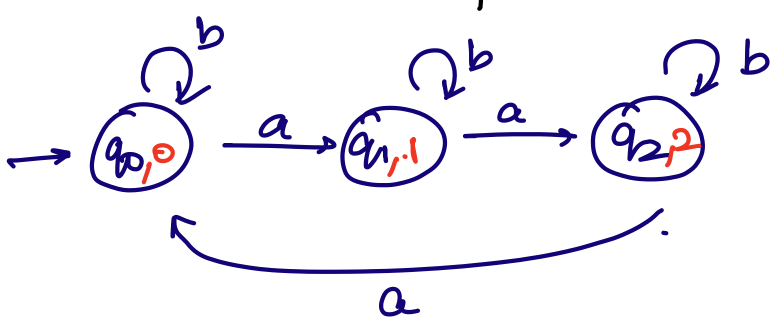
ab a abbbab

$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_1 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{b} q_0 \xrightarrow{b} q_0 \xrightarrow{a} q_1\ 0 \xrightarrow{b} q_2\ 1$

0 0 1 0 0 1 0 0

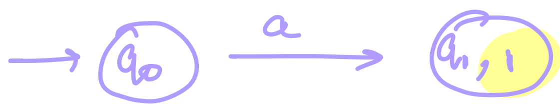
Convert Moore to mealy machine

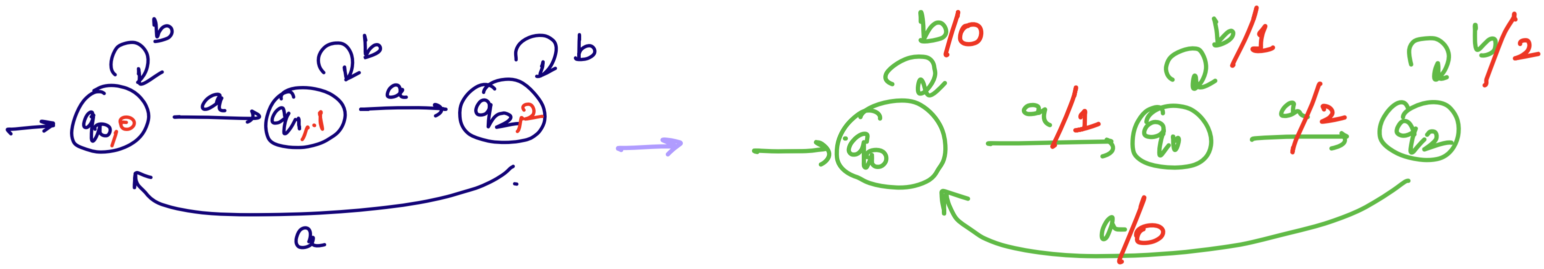
Eg:



count a%3

output → state associated
→ transition move



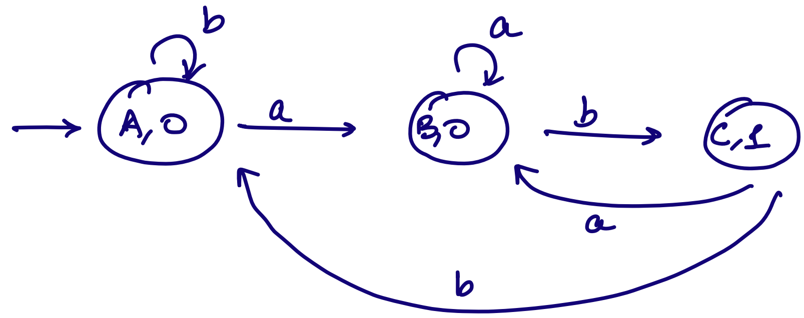


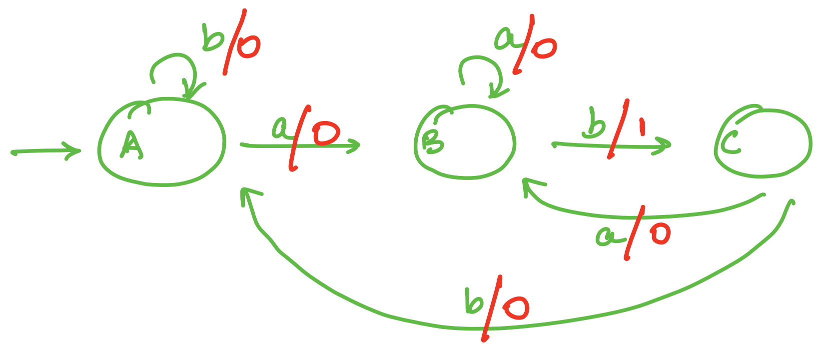
moore m/c

	a	b	Δ
q_0	q_1	q_0	0
q_1	q_2	q_1	1
q_2	q_0	q_2	2

	a	b
q_0	$q_1/1$	$q_0/0$
q_1	$q_2/2$	$q_1/1$
q_2	$q_0/0$	$q_2/2$

Q:

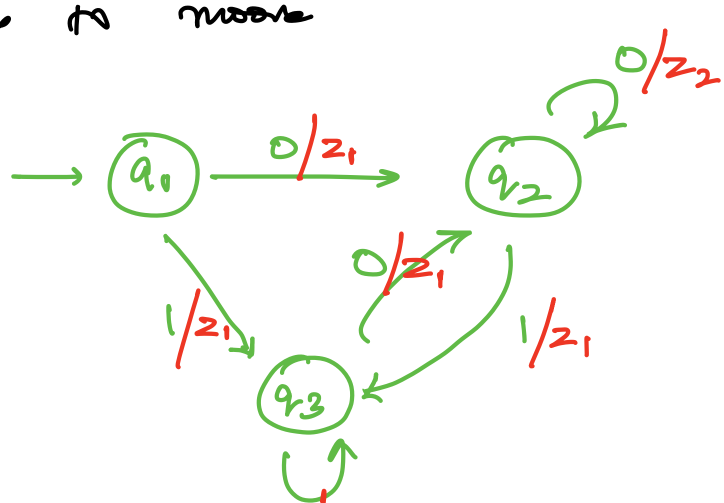




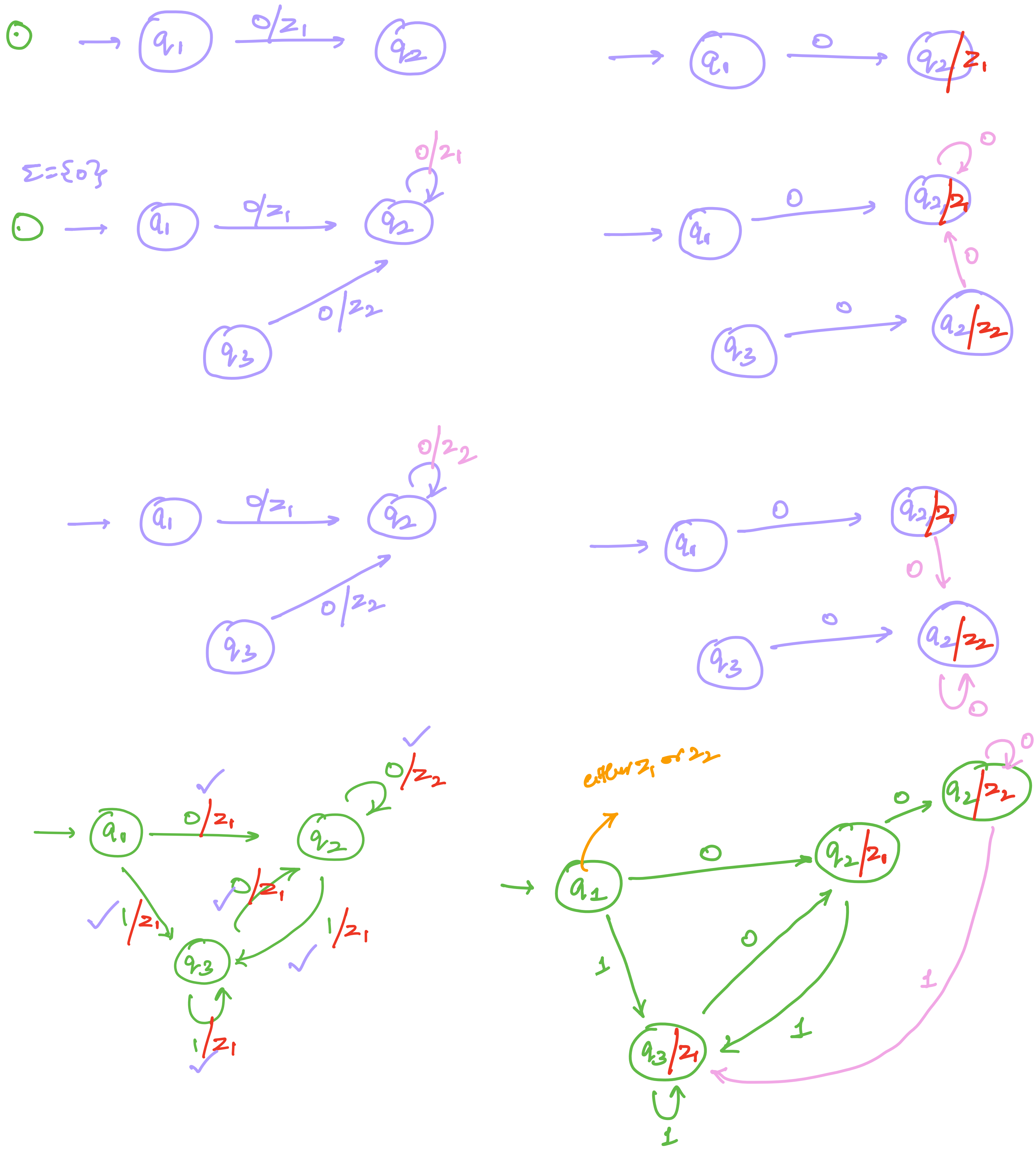
	a	b	Δ
A	B	A	0
B	B	C	0
C	B	A	1

	a	b
A	B/0	A/0
B	B/0	C/1
C	B/0	A/0

Convert mealy to moore



$1/z_1$



moore → mealy   no. of states same

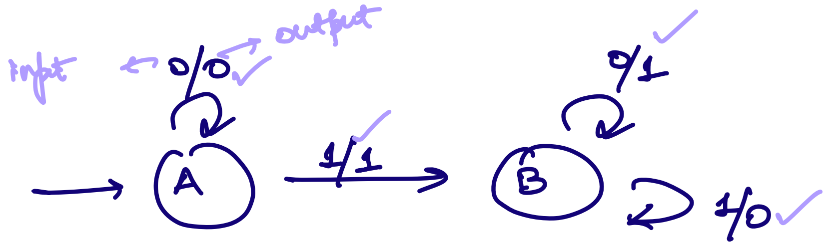
mealy → moore

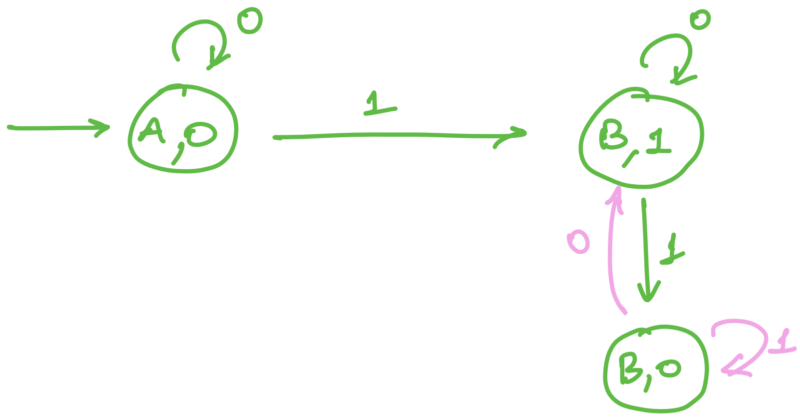
N → no of states

M → no of outputs

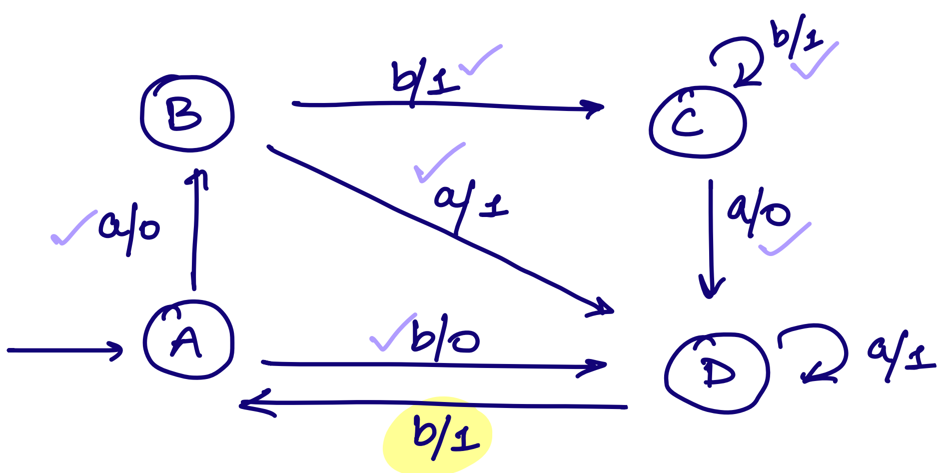
N*M are max no of states

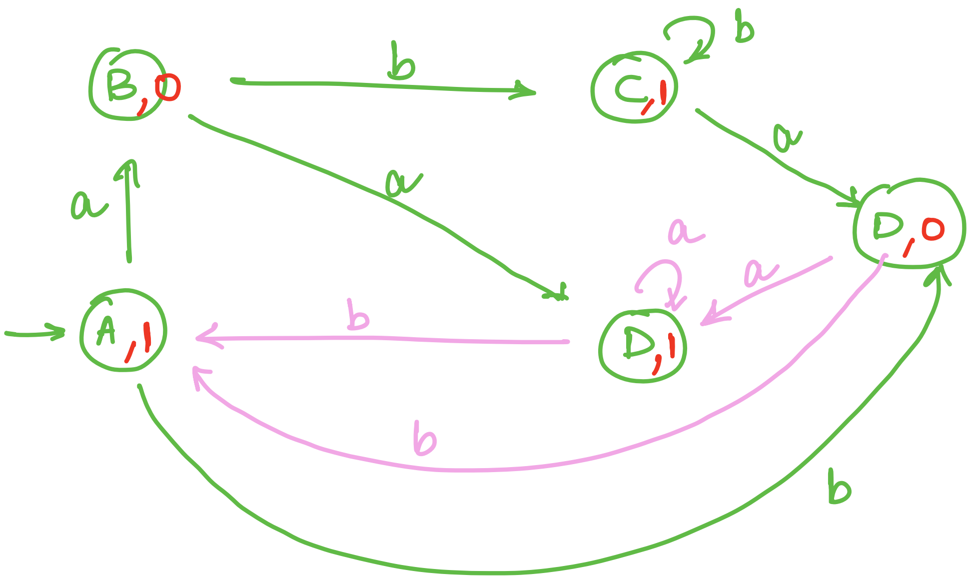
Eg:





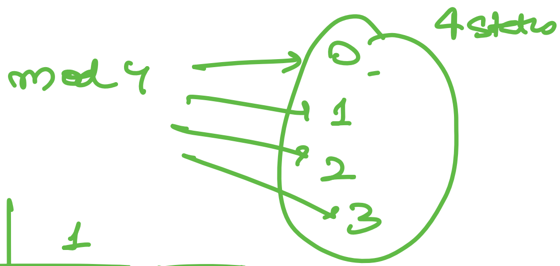
Q:





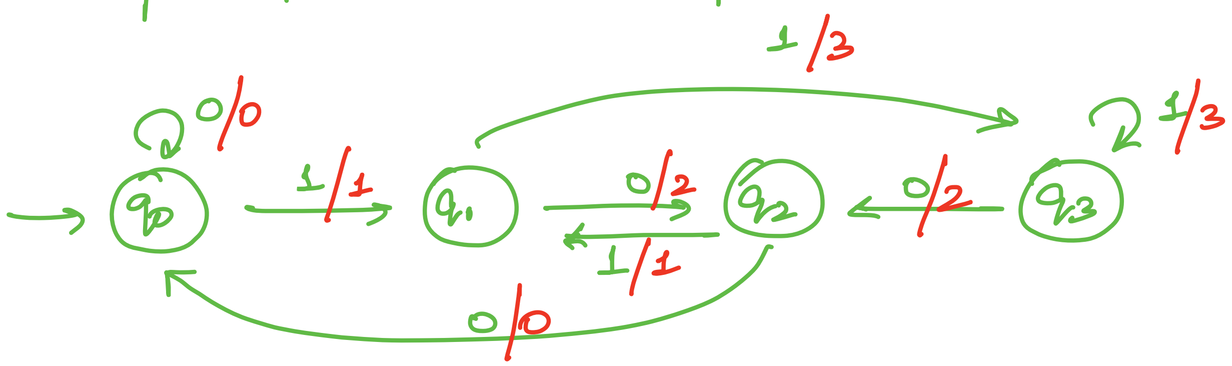
Q: Mealy machine which reads input string from $(0+1)^*$ and produces residue mod 4 for each binary string treated as binary integer.

16 8 4 2 1
$11010 = 26 \% 4 = 2$

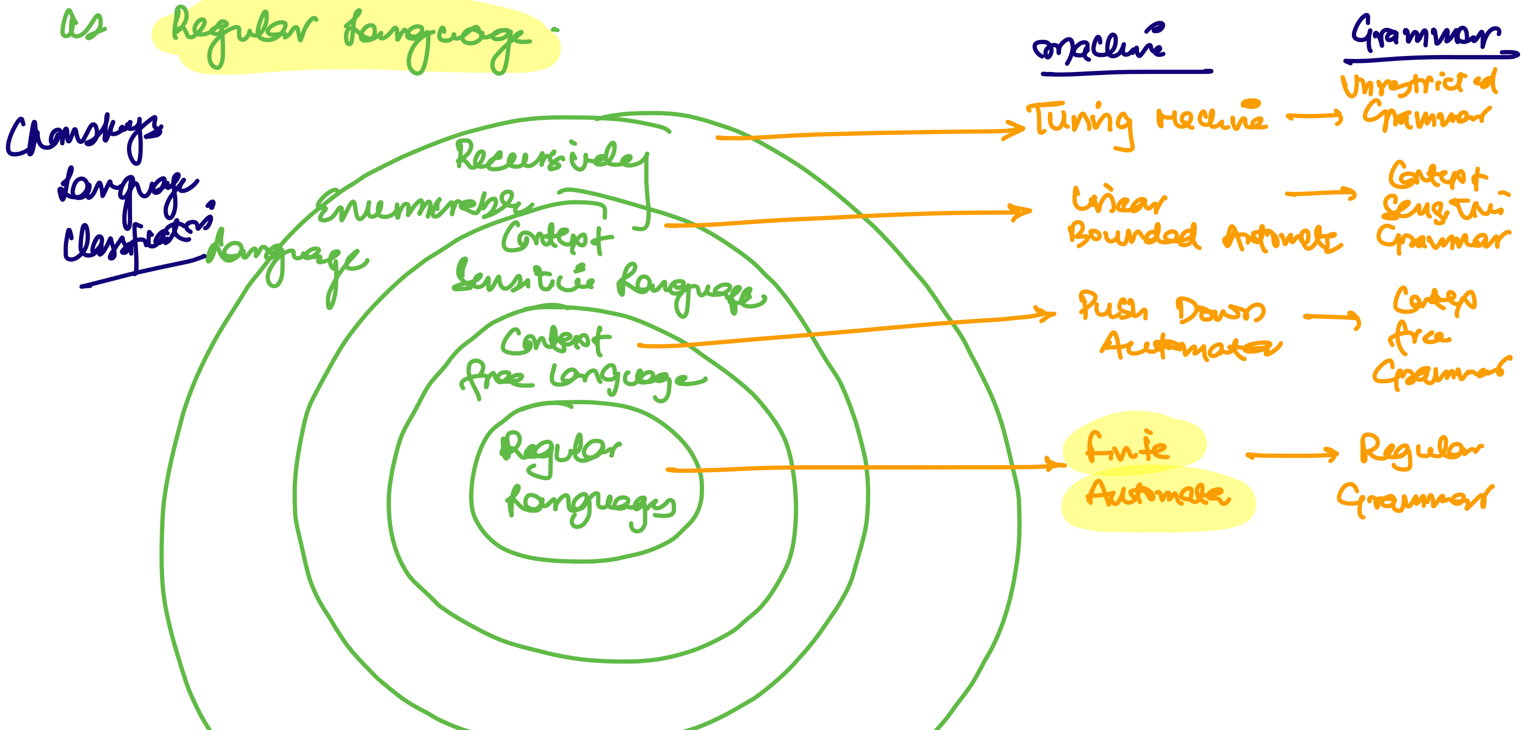


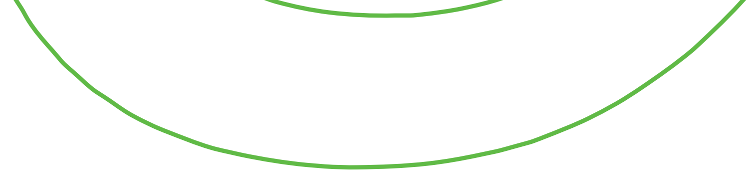
	0	1	
q_0	q_0	q_1	0
q_1	q_2	q_3	1
q_2	q_0	q_1	2
q_3	q_2	q_3	3

	0	1
q_0	$q_0/0$	$q_1/1$
q_1	$q_2/2$	$q_3/3$
q_2	$q_0/0$	$q_1/1$
q_3	$q_2/2$	$q_3/3$



Language which is accepted by FA is called as Regular language.





Regular Expression

↑ Representation

Regular Language

- Generator → Regular Grammar
  - Right Linear Grammar
  - Left Linear Grammar
- Acceptor → Finite Automata
  - without o/p
    - DFA
    - NFA
    - εNFA
  - with o/p
    - Moore
    - Mealy

Regular Expression:

↳ Representation of language accepted by FA.

3 operators:

i) + (union) $a+b$

ii) . (concatenation) $a \cdot b$

iii) * (kleene closure) $a^* = \{\varepsilon, a, aa, aaa, aaaa \ldots\}$

a) Primitive RE

ϕ, ε, Σ (input alphabet)

ϕ, ϵ, a, b

b) r_1, r_2 RE

$r_1 + r_2 \quad r_1 \cdot r_2 \quad r_1^*$ are RE

c) Can apply (a) and (b) as many times as you want.

$a + b$

$(a+b)\cdot a$

$(a+b)\, a \cdot b$

$((a+b)a\cdot b)^*$

$\Big\}$ RE.

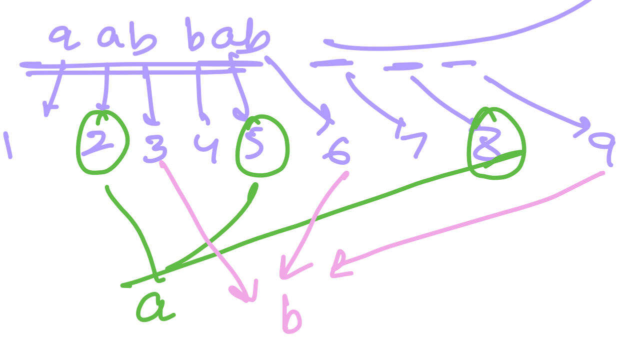
$\rightarrow$ aabab ✗

aab, aab bab

$((a+b)ab)^2$

$((a+b)ab)((a+b)ab)$

length % 3 = 0



$(a+b)^* = \Sigma^*$

baa

$\underline{(a+b)}\ \underline{(a+b)}\ \underline{(a+b)}$

b a a

RE

begins with a : $a(a+b)^*$

ends with a : $(a+b)^* a$

substring a : $(a+b)^* a (a+b)^*$

begins with ab : $ab(a+b)^*$

ends ab : $(a+b)^* ab$

substring ab : $(a+b)^* ab (a+b)^*$

$a^+ = a \cdot a^* = \{a, aa, aaa, aaaa \ldots\}$