

1/5/21

## Unit - 1

### Natural Language :-

- \* The language used to communicate with others.
  - \* It is highly flexible.
  - \* It is powerful.
  - \* No special ~~test~~ effort is needed to learn it.
  - \* It is ambiguous (confusion).
- ex: Hindi, English, Telugu etc.

### Formal Language :-

- \* They have strict rules and values to be followed.
- \* They have semantics and syntaxes which are predefined.
- \* It is unambiguous.
- \* It is designed to communicate with machine.

### Automata Theory :-

- \* It is used to implement Formal language.
- \* It is a mathematical model for automata theory.

### Applications of Formal Languages :-

- ① Text processing.
- ② Compiler design.
- ③ Hardware design.

### Alphabet :-

An alphabet is non-empty and finite set of symbols.

Ex:  $X = \{0, 1\}$  is alphabet.

$Y = \{0, 1, 2, \dots\}$  is not alphabet.

### String :-

It is a finite sequence of symbols from some alphabet.

Ex: "xyz" is string over alphabet  $\Sigma = \{a, b, c, \dots, x, y, z\}$ .

Language :-

Language is a collection of strings.

Ex: ①  $L_1 = \{01, 0011, 000111\}$  is a language over alphabet  $\{0, 1\}$ .

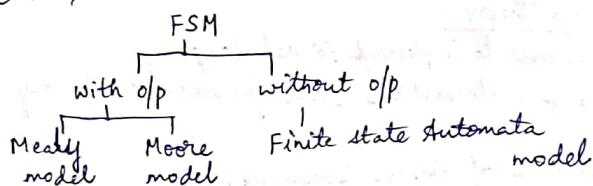
②  $L_2 = \{\epsilon, 0, 00, 000, \dots\}$  is a language over alphabet  $\{\epsilon, 0\}$ .

③  $L_3 = \{0^n 1^n 2^n : n \geq 1\}$  is a language over alphabet  $\{0, 1, 2\}$ .  
 ↳ for  $n=1$     012  
 for  $n=2$     001122  
 for  $n=3$     000111222    ( $\epsilon$  is empty string).

Finite-state Machine :-

The finite state system represents a mathematical model of a system with some input. The model finally gives some output.

Ex: Elevator, switch



A finite automaton is formerly defined as a 5-tuple  $(Q, \Sigma, \delta, q_0, F)$  where, (for DFA).

$Q$  is a finite set of states which is non-empty.

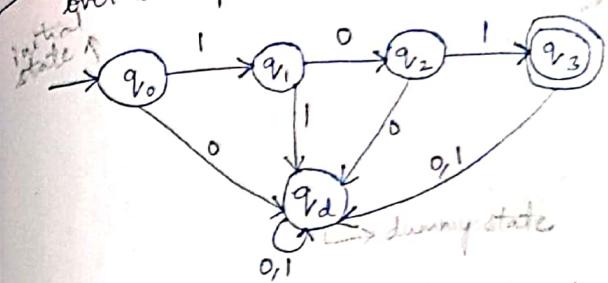
$\Sigma$  is the input alphabet

$q_0$  is the initial state

$F$  is a set of final states and  $F \subseteq Q$ .

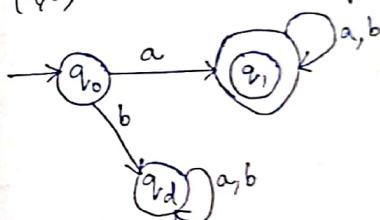
$\delta$  is a transition function or mapping function  $Q \times \Sigma \rightarrow Q$  using this the next state can be determined depending on the current input.

Q1: Design the DFA which accepts the string 101 over the input 0, 1.



$$\begin{array}{l|l}
 Q = \{q_0, q_1, q_2, q_3, q_d\} & \delta(q_0, 1) \rightarrow q_1 \\
 \Sigma = \{0, 1\} & \delta(q_0, 0) \rightarrow q_d \\
 q_0 = q_0 & \delta(q_1, 0) \rightarrow q_2 \\
 F = q_3 & \delta(q_1, 1) \rightarrow q_d \\
 & \delta(q_2, 1) \rightarrow q_3 \\
 & \delta(q_3, 1) \rightarrow q_d, \delta(q_3, 0) \rightarrow q_d, \delta(q_2, 0) \rightarrow q_d \\
 & \delta(q_d, 0) \rightarrow q_d, \delta(q_d, 1) \rightarrow q_d
 \end{array}$$

Q2: Construct a DFA which accepts the string over  $(a, b)$  and all the strings starts with "a".



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

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

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

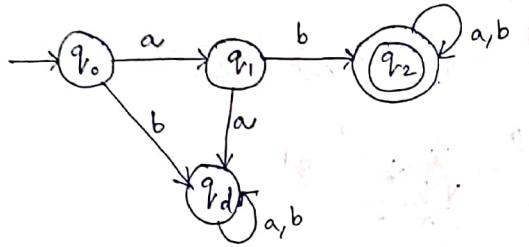
$$\delta(q_1, a) \rightarrow q_1, \delta(q_1, b) \rightarrow q_d$$

$$\delta(q_d, a) \rightarrow q_d, \delta(q_d, b) \rightarrow q_d$$

$$q_0 = q_0$$

$$F = q_1$$

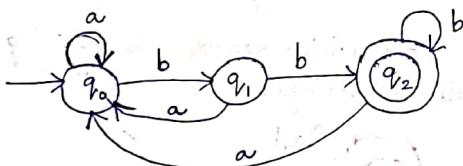
Q3. Construct a DFA which accepts the string over  $\{a, b\}$  and all strings start with "ab".



H.W.

- Design a DFA which accepts the string which ends with bb over the inputs  $\{a, b\}$ .
- Design a DFA that accepts the string containing ab as substring over the input  $\{a, b\}$ .

①



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

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

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

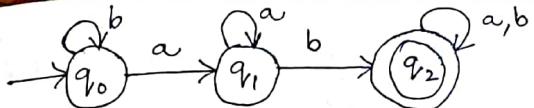
$$\delta(q_1, a) \rightarrow q_0, \delta(q_1, b) \rightarrow q_2$$

$$\delta(q_2, a) \rightarrow q_0, \delta(q_2, b) \rightarrow q_2$$

$$q_0 = q_0$$

$$F = q_2$$

②



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

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

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

$$\delta(q_1, a) \rightarrow q_1, \delta(q_1, b) \rightarrow q_2$$

$$\delta(q_2, a) \rightarrow q_2, \delta(q_2, b) \rightarrow q_d$$

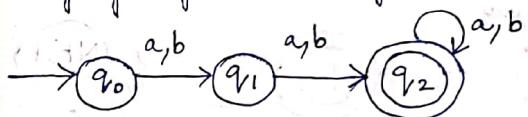
$$q_0 = q_0$$

$$F = q_2$$

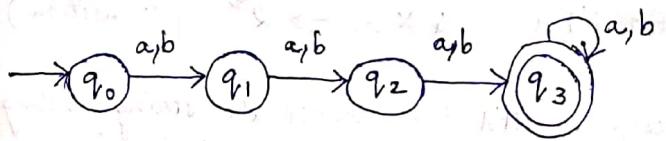
H.W.

- Design a DFA which takes the length of string  $\geq 2$  over input alphabet  $\{a, b\}$ .
- Design a DFA which takes the length of string  $\geq 3$  over input alphabet  $\{a, b\}$ .
- Construct a DFA which accepts all the strings of length divisible by 3 over  $\{a, b\}$ .

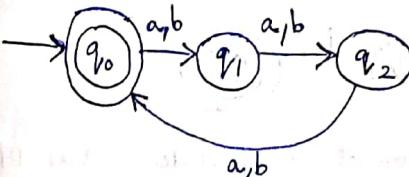
①



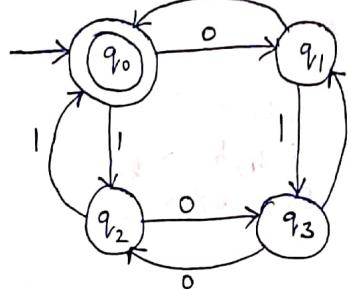
②



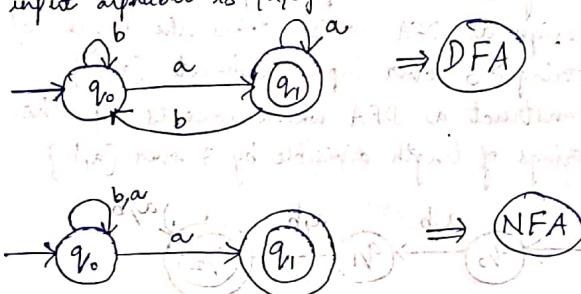
③



Q1: Design a DFA that accepts the string which contains an even no. of 0's and even no. of 1's.

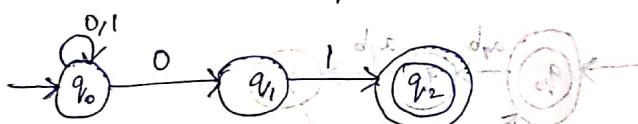


Q2: Design a NFA which ends with "a" where the input alphabet is {a, b}.



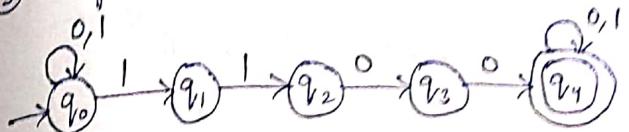
For NFA,  $Q \times \Sigma \rightarrow 2^{\Sigma}$  (transition).

Q3: Design a NFA that access the string ending with "01" where input alphabet is {0, 1}.



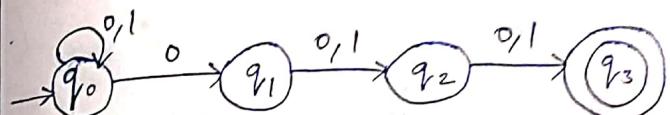
No need of transition at every state unlike DFA.

Q4: Design a NFA that contains "1100" as substring.

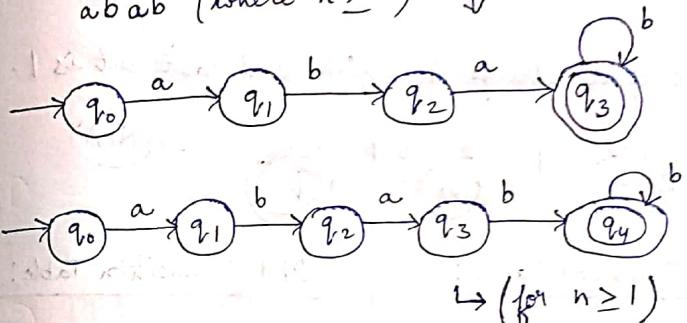


H.W:-

① Design a NFA with input alphabet {0, 1} where the third symbol from right is 0.

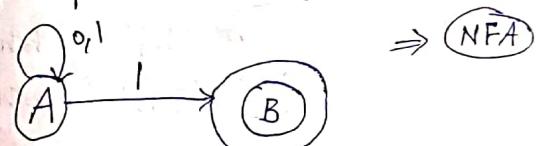


Q5: ① Construct a NFA accepting the language  $abab^n$  (where  $n \geq 0$ ).



Conversion of NFA to DFA :-

① Set of all strings that ends with 1 where the input alphabets are {0, 1}.



NFA Transition Table:

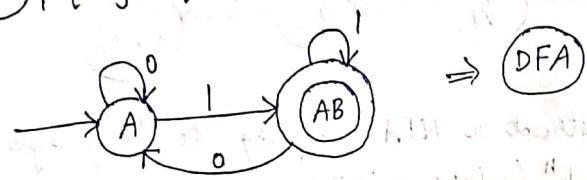
	0	1
$\rightarrow A$	$\{A\}$	$\{A, B\}$
$(B)$	$\emptyset$	$\emptyset$

DFA Transition Table:

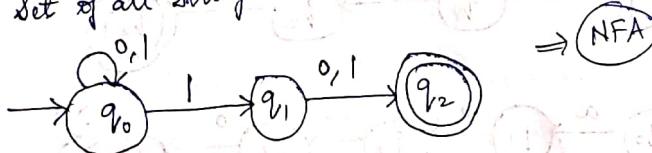
Same as NFA

	0	1
$\rightarrow A$	$\{A\}$	$\{AB\}$
$(AB)$	$\{A\}$	$\{AB\}$

$$\begin{aligned} \because S(AB, 0) &= S(A, 0) \cup S(B, 0) \\ S(AB, 1) &= S(A, 1) \cup S(B, 1) \end{aligned}$$



② Set of all strings in which 2nd last bit is 1.



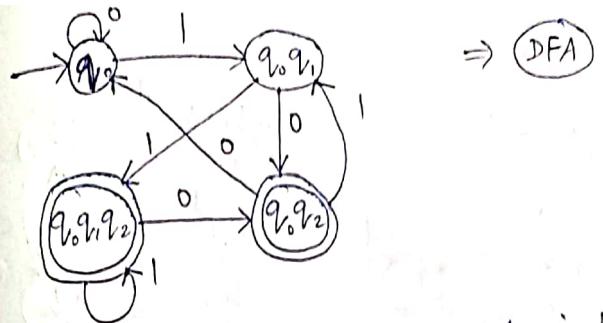
NFA Transition Table:

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

$\downarrow q_2$  is final here  
So all combinations of  $q_2$  will be final here

DFA Transition Table:

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



\* No. of states in DFA  $\geq$  No. of states in NFA

\* Max. no. of states in DFA =  $2^{\text{no. of states in NFA}}$

Equivalence of 2 finite state machines:

H.W.

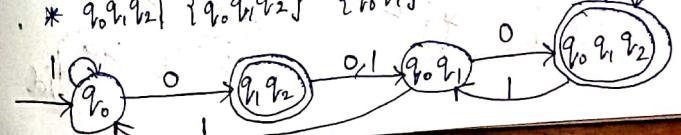
① Convert the following NFA to a DFA.

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

final state

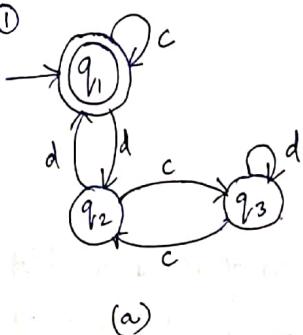
Sol.: DFA Transition Table:

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

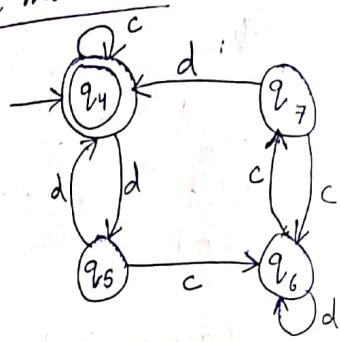


Equivalence of 2 finite state machines :-

Q. ①



(a)



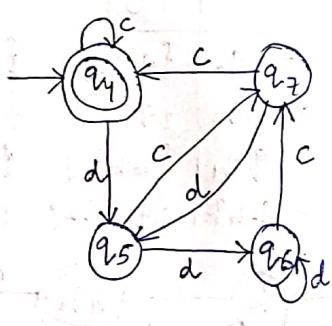
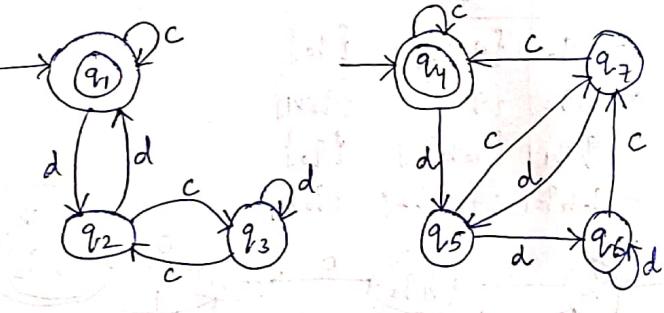
(b)

Are 'a' and 'b' machines equivalent?

soln:	States	c	d	i - int
	$(q_1, q_4)$	$(q_1, q_4)$	$(q_2, q_5)$	f - final
		f f	i i	
	$(q_2, q_5)$	$(q_3, q_6)$	$(q_1, q_4)$	
	i i	f f		
	$(q_3, q_6)$	$(q_2, q_7)$	$(q_3, q_6)$	
	i i	i i	i i	
	$(q_2, q_7)$	$(q_3, q_6)$	$(q_1, q_4)$	
	i i	f f		

Hence the given machines are equivalent.

②



soln: states

$(q_1, q_4)$

f f

$(q_2, q_5)$

i i

$(q_3, q_6)$

f i

$(q_1, q_6)$

f i

c  
 $(q_1, q_4)$

f f

$(q_2, q_5)$

i i

$(q_3, q_6)$

f i

d  
 $(q_2, q_5)$

i i

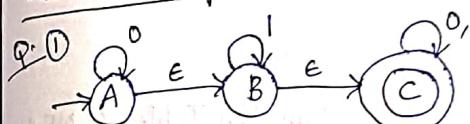
$(q_1, q_6)$

f i

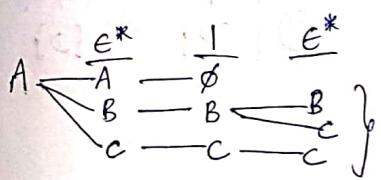
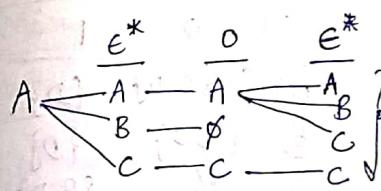
Upon giving input d to  $q_2$  it moves to final state  $q_1$ , but  $q_5$  moves to intermediate state  $q_6$ .

Hence the given machines are not equivalent.

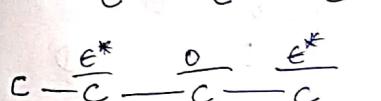
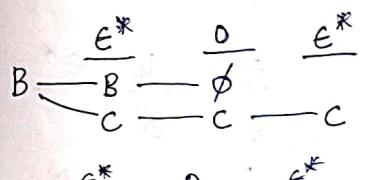
Conversion of  $\epsilon$ -NFA to NFA :-

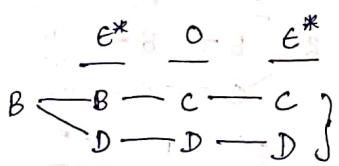
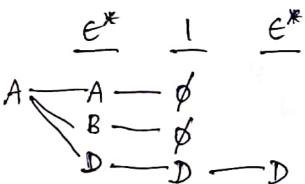
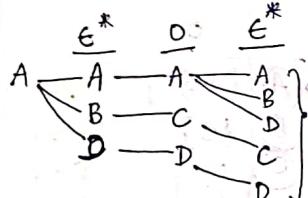
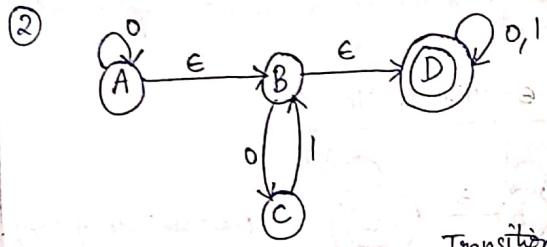
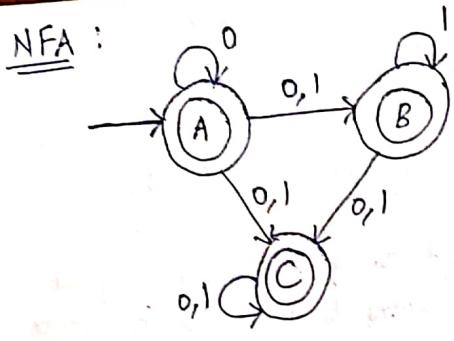


Transition Table of NFA



Any state which reaches final state with  $\epsilon$  is also final state.





Transition Table of NFA :-

	0	1
A	{A, B, C, D}	{D}
B	{C, D}	{D}
C	∅	{B, D}
D	{D}	{D}

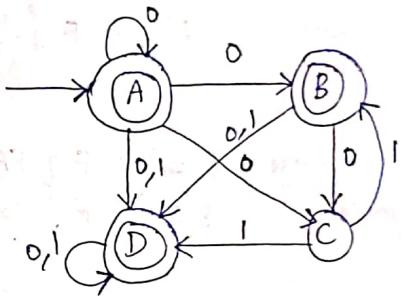
c - c - φ      φ

D - D - D - D

c - c - B - B      D

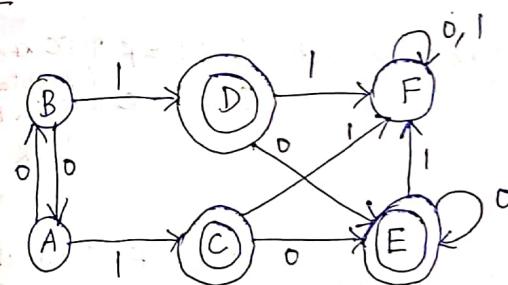
D - D - D - D

NFA :



Minimisation of DFA :-

Method 1: δ MYHILL NERODE (Table filling method)



A B C D E F

A						
B						
C	✓	✓				
D	✓	✓				
E	✓	✓				
F						✓

put ✓ when  
1 state is final  
& other is not.

Now consider the unchecked boxes.

$$\begin{array}{l} \text{BA} \\ \hline \begin{array}{l} \delta(B, 0) = A \\ \delta(A, 0) = B \end{array} \left. \begin{array}{l} \text{AB is unchecked} \\ \text{so ignore.} \end{array} \right\} \quad \begin{array}{l} \delta(B, 1) = D \\ \delta(A, 1) = C \end{array} \left. \begin{array}{l} \text{DC unchecked} \\ \text{ignore} \end{array} \right\} \end{array}$$

$$\begin{array}{l} \text{DC} \\ \hline \begin{array}{l} \delta(D, 0) = E \\ \delta(C, 0) = E \end{array} \left. \begin{array}{l} \text{EE doesn't exist.} \\ \text{do ignore.} \end{array} \right\} \quad \begin{array}{l} \delta(D, 1) = F \\ \delta(C, 1) = F \end{array} \left. \begin{array}{l} \text{FF doesn't} \\ \text{exist.} \\ \text{ignore} \end{array} \right\} \end{array}$$

$$\begin{array}{l} \text{EC} \\ \hline \begin{array}{l} \delta(E, 0) = E \\ \delta(C, 0) = D \end{array} \left. \begin{array}{l} \text{ED not there.} \\ \text{ignore..} \end{array} \right\} \quad \begin{array}{l} \delta(E, 1) = F \\ \delta(C, 1) = F \end{array} \left. \begin{array}{l} \text{FF not there.} \\ \text{so ignore.} \end{array} \right\} \end{array}$$

$$\begin{array}{l} \text{ED} \\ \hline \begin{array}{l} \delta(E, 0) = E \\ \delta(D, 0) = E \end{array} \left. \begin{array}{l} \text{EE not there.} \\ \text{ignore} \end{array} \right\} \quad \begin{array}{l} \delta(E, 1) = F \\ \delta(D, 1) = F \end{array} \left. \begin{array}{l} \text{FF not there} \\ \text{ignore} \end{array} \right\} \end{array}$$

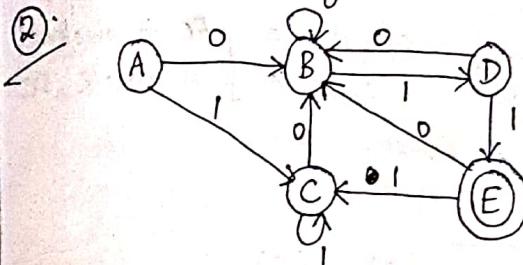
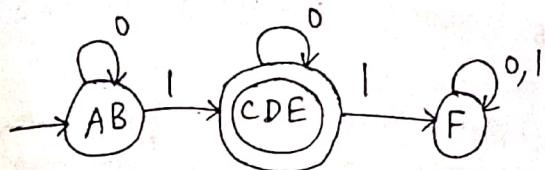
$$\begin{array}{l} \text{FA} \\ \hline \begin{array}{l} \delta(F, 0) = F \\ \delta(A, 0) = B \end{array} \left. \begin{array}{l} \text{FB is unchecked.} \\ \text{so ignore.} \end{array} \right\} \quad \begin{array}{l} \delta(F, 1) = F \\ \delta(A, 1) = C \end{array} \left. \begin{array}{l} \text{FC is checked} \\ \text{& obtained} \\ \text{from FA. So} \\ \text{check FA.} \end{array} \right\} \end{array}$$

$$\begin{array}{l} \text{FB} \\ \hline \begin{array}{l} \delta(F, 0) = F \\ \delta(B, 0) = A \end{array} \left. \begin{array}{l} \text{FA is checked.} \\ \text{so check FB.} \end{array} \right\} \quad \begin{array}{l} \delta(F, 1) = F \\ \delta(B, 1) = D \end{array} \left. \begin{array}{l} \text{FD checked.} \\ \text{Also FB is} \\ \text{checked.} \end{array} \right\} \end{array}$$

From table, unchecked states are,

$$\begin{array}{l} (\text{A}, \text{B}) \quad (\text{D}, \text{C}) \quad (\text{E}, \text{C}) \quad (\text{E}, \text{D}) \\ \qquad \qquad \qquad \xrightarrow{\text{can be combined}} \\ (\text{A}, \text{B}) \quad \overbrace{(\text{D}, \text{C}, \text{E})}^{\xrightarrow{\text{is added to}}} \quad (\text{F}) \xrightarrow{\text{include all states}} \end{array}$$

$\therefore (\text{A}, \text{B}) \quad (\text{C}, \text{D}, \text{E}) \quad (\text{F})$



	A	B	C	D	E
A					
B		✓			
C			✓		
D	✓	✓	✓		
E	✓	✓	✓	✓	✓

Now consider the unchecked states.

$$\begin{array}{l} \text{BA} \\ \hline \begin{array}{l} \delta(B, 0) = B \\ \delta(A, 0) = B \end{array} \left. \begin{array}{l} \text{BB} \\ \text{ignore.} \end{array} \right\} \quad \begin{array}{l} \delta(B, 1) = D \\ \delta(A, 1) = C \end{array} \left. \begin{array}{l} \text{DC unchecked} \\ \text{ignore.} \end{array} \right\} \end{array}$$

$$\begin{array}{l} \text{CA} \\ \hline \begin{array}{l} \delta(C, 0) = B \\ \delta(A, 0) = B \end{array} \left. \begin{array}{l} \text{BB} \\ \text{ignore} \end{array} \right\} \quad \begin{array}{l} \delta(C, 1) = C \\ \delta(A, 1) = C \end{array} \left. \begin{array}{l} \text{CC} \\ \text{ignore.} \end{array} \right\} \end{array}$$

$$\begin{array}{l} \text{CB} \\ \hline \begin{array}{l} \delta(C, 0) = B \\ \delta(B, 0) = B \end{array} \left. \begin{array}{l} \text{BB} \\ \text{ignore} \end{array} \right\} \quad \begin{array}{l} \delta(C, 1) = C \\ \delta(B, 1) = D \end{array} \left. \begin{array}{l} \text{CD} \\ \text{ignore.} \end{array} \right\} \end{array}$$

$$\begin{array}{l} \text{DA} \\ \overline{\delta(D, 0) = B} \quad \left. \begin{array}{l} \text{BB} \\ \text{ignore} \end{array} \right\} \\ \overline{\delta(A, 0) = B} \end{array}$$

$$\begin{array}{l} \left. \begin{array}{l} \delta(D, 1) = E \\ \delta(A, 1) = C \end{array} \right\} \text{EC is checked} \\ \text{So check } \delta_C \end{array}$$

$$\begin{array}{l} \text{DS} \\ \overline{\delta(D, 0) = B} \quad \left. \begin{array}{l} \text{BB} \\ \text{ignore} \end{array} \right\} \\ \overline{\delta(B, 0) = B} \end{array}$$

$$\begin{array}{l} \left. \begin{array}{l} \delta(D, 1) = E \\ \delta(B, 1) = D \end{array} \right\} \text{ED is checked} \\ \text{So check } \delta_E \end{array}$$

$$\begin{array}{l} \text{DC} \\ \overline{\delta(D, 0) = B} \quad \left. \begin{array}{l} \text{BB} \\ \text{ignore} \end{array} \right\} \\ \overline{\delta(C, 0) = B} \quad \text{(cont.)} \end{array}$$

$$\begin{array}{l} \left. \begin{array}{l} \delta(D, 1) = E \\ \delta(C, 1) = C \end{array} \right\} \text{EC is checked} \\ \text{So check } \delta_C \end{array}$$

From table, unchecked states are,

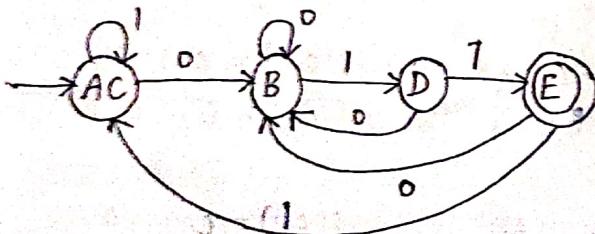
$$\times \underbrace{(E, A) \quad (C, A) \quad (C, B)}_{(A, B, C)} \quad (D) \quad (E)$$



From transition table,  
AC on input 1 goes to C but  
B goes to D. So ABC do  
not form a state.

Hence split (A, C), (B). X

$$\therefore (A, C) \quad (B) \quad (D) \quad (E)$$



### Procedure / Steps :-

1. Draw a table for all pairs of states (P, Q).
2. Mark all pairs where P ∈ F and Q ∉ F.
3. If there are any unmarked pairs (P, Q) such that  $[\delta(P, x), \delta(Q, x)]$  is marked, then mark [P, Q] where 'x' is an input symbol.

Repeat this until no more markings can be done.

4. Combine all the unmarked pairs and make them a single state in the minimized DFA.

(In the exam, first write these steps and then solve the given problem).

(cont.) . . .

Now repeat the procedure for unchecked states.

$$\begin{array}{l} \text{BA} \\ \overline{\delta(B, 0) = B} \quad \left. \begin{array}{l} \text{BB} \\ \text{ignore} \end{array} \right\} \\ \overline{\delta(A, 0) = B} \end{array} \quad \begin{array}{l} \delta(B, 1) = D \quad \left. \begin{array}{l} \text{DC is checked} \\ \text{So check } \delta_B \end{array} \right\} \\ \delta(A, 1) = C \end{array}$$

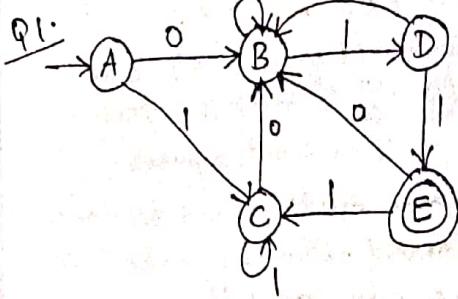
$$\begin{array}{l} \text{CA} \\ \overline{\delta(C, 0) = B} \quad \left. \begin{array}{l} \text{BB} \\ \text{ignore} \end{array} \right\} \\ \overline{\delta(A, 0) = B} \end{array} \quad \begin{array}{l} \delta(C, 1) = C \quad \left. \begin{array}{l} \text{CC ignore} \\ \delta(A, 1) = C \end{array} \right\} \end{array}$$

$$\begin{array}{l} \text{CB} \\ \overline{\delta(C, 0) = B} \quad \left. \begin{array}{l} \text{BB} \\ \text{ignore} \end{array} \right\} \\ \overline{\delta(B, 0) = B} \end{array} \quad \begin{array}{l} \delta(C, 1) = C \quad \left. \begin{array}{l} \text{CD is checked} \\ \delta(B, 1) = D \end{array} \right\} \\ \text{So check } \delta_C \end{array}$$

∴ Now, from table,  
(A, C) (B) (D) (E)

and then the minimized DFA

### Method 2: Equivalence Method.



0-Equivalence divide by final & non-final states

$\{A, B, C, D\} \setminus \{E\}$

1-Equivalence refer from 0 equi.

$\{A, B, C\} \setminus \{D\} \setminus \{E\}$

2-Equivalence refer from 1 equi.

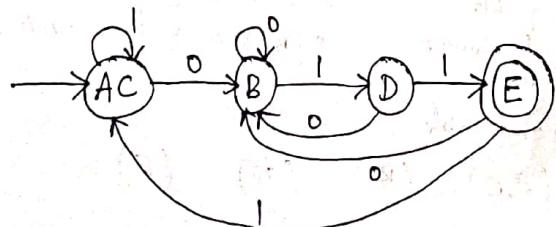
$\{A, C\} \setminus \{B\} \setminus \{D\} \setminus \{E\}$

3-Equivalence refer from 2 equi.

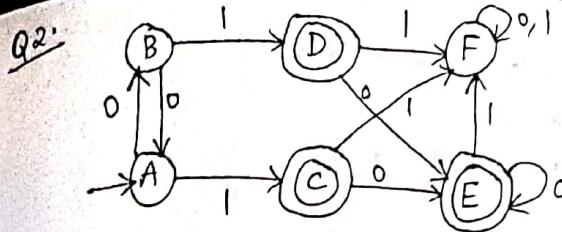
$\{A, C\} \setminus \{B\} \setminus \{D\} \setminus \{E\}$

Now the last two equivalences (here 2, 3) became equal.  
Hence we can stop here.

$\therefore (A, C), (B), (D), (E)$ .



Splitting based on inputs & corr. transitions leading to same group or not. (Like partition table STD).



0-Equivalence

$\{A, B, F\} \setminus \{C, D, E\}$

1-Equivalence

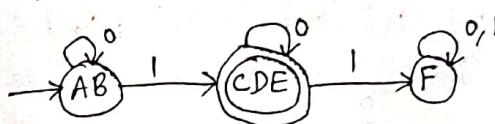
$\{A, B\} \setminus \{F\} \setminus \{C, D, E\}$

2-Equivalence

$\{A, B\} \setminus \{F\} \setminus \{C, D, E\}$

Now we can stop.

$\therefore (A, B), (C, D, E), (F)$



③ Construct the minimised DFA by using the two methods.

	0	1
$\rightarrow q_0$	$q_1$	$q_5$
$q_1$	$q_6$	$q_2$
$q_2$	$q_0$	$q_2$
$q_3$	$q_2$	$q_0$
$q_4$	$q_7$	$q_5$
$q_5$	$q_2$	$q_6$
$q_6$	$q_6$	$q_4$
$q_7$	$q_5$	$q_2$

### Equivalence Method :-

Transition table is given in q.s.

0-Equivalence

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

1-Equivalence

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

2-Equivalence

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

3-Equivalence

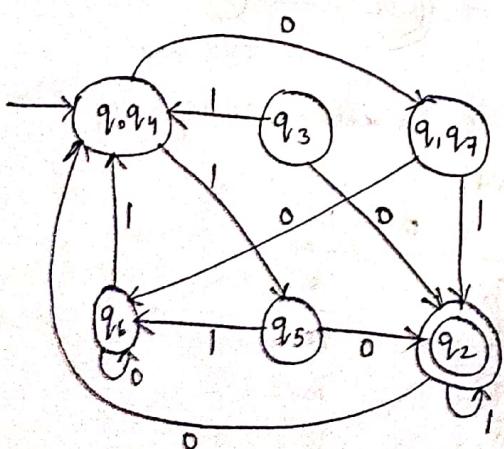
$$\{q_0, q_4\} \{q_6\} \{q_3\} \{q_5\} \{q_1, q_7\} \{q_2\}$$

4-Equivalence

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

so stop here.

$$\therefore (q_0, q_4) (q_6) (q_3) (q_5) (q_1, q_7) (q_2)$$



### My Hill Nerode Method :-

	$q_0$	$q_1$	$q_2$	$q_3$	$q_4$	$q_5$	$q_6$	$q_7$
$q_1$	✓							
$q_2$	✓	✓	✓					
$q_3$	✓	✓	✓	✓				
$q_4$		✓	✓	✓	✓			
$q_5$	✓	✓	✓	✓	✓	✓		
$q_6$	✓	✓	✓	✓	✓	✓	✓	
$q_7$	✓	✓	✓	✓	✓	✓	✓	✓

Consider the unchecked states.

$$\begin{array}{l} \frac{q_1 q_0}{\delta(q_1, 0)} = q_6 \\ \delta(q_0, 0) = q_1 \end{array} \left. \begin{array}{l} \text{Ignore} \\ q_1 \end{array} \right\} \quad \begin{array}{l} \delta(q_1, 1) = q_2 \\ \delta(q_0, 1) = q_5 \end{array} \left. \begin{array}{l} \text{Check} \\ q_2, q_5 \end{array} \right\}$$

$$\begin{array}{l} \frac{q_3 q_0}{\delta(q_3, 0)} = q_2 \\ \delta(q_0, 0) = q_1 \end{array} \left. \begin{array}{l} \text{Check} \\ q_2, q_1 \end{array} \right\} \quad \begin{array}{l} \delta(q_3, 1) = q_0 \\ \delta(q_0, 1) = q_5 \end{array} \left. \begin{array}{l} \text{Ignore} \\ q_0, q_5 \end{array} \right\}$$

$$\begin{array}{l} \frac{q_3 q_1}{\delta(q_3, 0)} = q_2 \\ \delta(q_1, 0) = q_6 \end{array} \left. \begin{array}{l} \text{Check} \\ q_2, q_6 \end{array} \right\} \quad \begin{array}{l} \delta(q_3, 1) = q_0 \\ \delta(q_1, 1) = q_2 \end{array} \left. \begin{array}{l} \text{Check} \\ q_0, q_2 \end{array} \right\}$$

$$\begin{array}{l} \frac{q_4 q_0}{\delta(q_4, 0)} = q_7 \\ \delta(q_0, 0) = q_1 \end{array} \left. \begin{array}{l} \text{Ignore} \\ q_1 \end{array} \right\} \quad \begin{array}{l} \delta(q_4, 1) = q_5 \\ \delta(q_0, 1) = q_5 \end{array} \left. \begin{array}{l} \text{Ignore} \\ q_5 \end{array} \right\}$$

$$\begin{array}{l} \frac{q_4 q_1}{\delta(q_4, 0)} = q_7 \\ \delta(q_1, 0) = q_6 \end{array} \left. \begin{array}{l} \text{Ignore} \\ q_6 \end{array} \right\} \quad \begin{array}{l} \delta(q_4, 1) = q_5 \\ \delta(q_1, 1) = q_2 \end{array} \left. \begin{array}{l} \text{Check} \\ q_5, q_2 \end{array} \right\}$$

$$\left. \begin{array}{l} q_4 q_3 \\ \delta(q_4, 0) = q_1 \\ \delta(q_3, 0) = q_2 \end{array} \right\} \text{check } q_4 q_3$$

$$\begin{aligned} \frac{q_5 q_0}{\delta(q_5, 0)} &= q_2 \\ \delta(q_0, 0) &= q_1 \end{aligned} \quad \left. \begin{array}{l} \text{check} \\ q_5 q_0 \end{array} \right\}$$

$$\begin{aligned} & \frac{q_5 q_1}{\delta(q_5, 0)} = q_2 \\ & \delta(q_1, 0) = q_6 \end{aligned} \quad \left. \begin{array}{l} \text{check} \\ q_5 q_1 \end{array} \right.$$

$$\begin{aligned} & \frac{q_1 q_3}{\delta(2,5,0) = q_2} \\ & \delta(7,3,0) = q_2 \end{aligned} \quad \left. \right\} \text{Ignore}$$

$$\begin{aligned} \frac{q_5 q_4}{\delta(q_5, 0)} &= q_2 \\ \frac{q_5 q_4}{\delta(q_4, 0)} &= q_7 \end{aligned} \quad \left. \begin{array}{l} \text{check} \\ q_5 q_4 \end{array} \right.$$

$$\begin{aligned} \frac{q_6 q_0}{\delta(q_6, 0)} &= q_6 \\ \frac{q_8 q_0}{\delta(q_8, 0)} &= q_1 \end{aligned} \quad \left. \right\} \text{Ignore}$$

$$\begin{aligned} \frac{q_6 q_1}{\delta(q_6, 0)} &= q_6 \\ \delta(q_1, 0) &= q_6 \end{aligned} \} \text{Ignore}$$

$$\begin{aligned} \underline{s(q_6, 0) = q_6} \\ s(q_3, 0) = q_2 \end{aligned} \quad \left. \begin{array}{l} \text{Check} \\ q_6q_3 \end{array} \right\}$$

$$\left. \begin{array}{l} \delta(q_4, 1) = q_5 \\ \delta(q_3, 1) = q_0 \end{array} \right\} \text{Eq. 1}$$

$$\begin{aligned} \delta(\varphi_5, 1) &= v_6 \\ \delta(\varphi_6, 1) &= v_5 \end{aligned} \quad \left. \right\} g_{g^{(n+1)}}$$

$$\left. \begin{array}{l} \delta(q_5, 1) = q_6 \\ \delta(q_1, 1) = q_2 \end{array} \right\} \begin{array}{l} \text{check} \\ q_{\leq 9} \end{array}$$

$$\begin{aligned} \delta(q_5, 1) &= q_6 \\ \delta(q_3, 1) &= q_0 \end{aligned} \quad \left. \right\} \text{ignores}$$

$$\begin{aligned} \delta(q_B, 1) &= q_6 \\ \delta(q_H, 1) &= q_5 \end{aligned} \quad \left. \right\} \text{Synch.}$$

$$\begin{aligned} \delta(q_0, 1) &= q_1 \\ \delta(q_0, 1) &= q_2 \end{aligned} \quad \left. \begin{array}{l} \text{check} \\ q_0 q_1 \\ q_0 q_2 \end{array} \right.$$

$$\begin{aligned} \delta(q_0, l) &= q_4 \\ \delta(q_1, l) &= q_2 \end{aligned} \quad \left. \begin{array}{l} \text{check} \\ q_0 q_1 \end{array} \right.$$

$$\begin{aligned} \delta(q_{41}, 1) &= q_4 \\ \delta(q_{31}, 1) &= q_0 \end{aligned} \quad \left. \right\} \text{Ignore}$$

$$\begin{aligned} \frac{q_1 q_4}{8}(q_6, 0) &= q_6 \\ \frac{q_1 q_4}{8}(q_{14}, 0) &= q_{14} \end{aligned} \quad \left. \begin{array}{l} \text{against} \\ \text{against} \end{array} \right\}$$

$$\begin{aligned} \frac{q_6 q_5}{\delta(q_6, 0)} &= q_6 \\ \delta(q_5, 0) &= q_2 \end{aligned} \quad \left. \begin{array}{l} \text{check} \\ q_6 q_5 \end{array} \right\}$$

$$\begin{aligned} \frac{q_1 q_0}{8(q_1, 0)} &= q_6 \\ q_6 &= q_1 \end{aligned} \quad \left. \begin{array}{l} \text{Check} \\ q_1 q_6 \end{array} \right\}$$

$$\begin{aligned} \overline{s(q_1, 0)} &= q_6 \\ s(q_1, 0) &= q_6 \end{aligned} \quad \left. \right\} \text{Igröte}$$

$$\begin{aligned} \underline{q_1 q_3} \\ q_1(q_1, 0) &= q_6 \\ q_3(q_3, 0) &= q_2 \end{aligned} \quad \left. \begin{array}{l} \text{check} \\ q_1 q_3 \end{array} \right\}$$

$$\begin{aligned} \frac{q_1 q_4}{8(q_1, 0)} &= q_6 \\ \frac{8(q_4, 0)}{8(q_4, 0)} &= q_7 \end{aligned} \quad \left. \begin{array}{l} \text{Ignore} \\ \text{Ignore} \end{array} \right\}$$

$$\begin{aligned} \frac{q_1 q_5}{\delta(q_1, 0)} &= q_6 \\ \frac{q_1 q_5}{\delta(q_5, 0)} &= q_1 \end{aligned} \quad \text{check } q_1 q_5$$

$$\begin{aligned} \underline{q(9_1, 0)} &= q_1 \\ q(9_1, 0) &= q_1 \end{aligned} \quad \left. \right\} \text{Ignore}$$

$$\begin{aligned} \delta(q_{41}, 1) &= q_4 \\ \delta(q_{41}, 1) &\approx q_5 \end{aligned} \quad \left. \begin{array}{l} \text{Pleck} \\ \text{Frob} \\ q_{41} \end{array} \right\}$$

$$\begin{aligned}\delta(q_6, 1) &= q_7 \quad \text{J shake} \\ \delta(q_8, 1) &= q_6 \quad \text{q. 6. 7. 8}\end{aligned}$$

$$\left. \begin{array}{l} \delta(q_{A_1}, 1) = q_2 \\ \delta(q_{B_1}, 1) = q_5 \end{array} \right\} \text{blank}$$

$$\begin{cases} \delta(q_1, 1) = q_2 \\ \delta(q_2, 1) = q_1 \end{cases} \quad \left\{ \text{Eqn 11} \right.$$

$$\begin{aligned} \delta(q_{11}, 1) &= q_2 && \text{clock} \\ \delta(q_{11}, 1) &= q_0 && q_1 q_6 \end{aligned}$$

$$\begin{aligned} \delta(q_1, 1) &= q_2 \\ \delta(q_4, 1) &= q_5 \end{aligned} \quad \left. \begin{array}{l} \text{check} \\ \text{no } q_4 \end{array} \right.$$

$$\begin{aligned} \delta(q_7, 1) &= q_2 \\ \delta(q_5, 1) &= q_6 \end{aligned} \quad \left. \begin{array}{l} \text{check} \\ q_1q_5 \end{array} \right.$$

$$\begin{aligned} \delta(q_7, 1) &= q_2 \\ \delta(q_6, 1) &= q_4 \end{aligned} \quad ] \quad \text{check}$$

Now repeat the same procedure:

$$\begin{array}{l} q_7 q_8 \\ \delta(q_7, 0) = q_7 \\ \delta(q_8, 0) = q_1 \end{array} \left. \begin{array}{l} \text{Ignore} \\ \text{Ignore} \end{array} \right\}$$

$$\begin{array}{l} \delta(q_7, 1) = q_5 \\ \delta(q_8, 1) = q_5 \end{array} \left. \begin{array}{l} \text{Ignore} \\ \text{Ignore} \end{array} \right\}$$

$$\begin{array}{l} q_5 q_6 \\ \delta(q_5, 0) = q_2 \\ \delta(q_6, 0) = q_2 \end{array} \left. \begin{array}{l} \text{Ignore} \\ \text{Ignore} \end{array} \right\}$$

$$\begin{array}{l} \delta(q_5, 1) = q_6 \\ \delta(q_6, 1) = q_1 \end{array} \left. \begin{array}{l} \text{Ignore} \\ \text{Ignore} \end{array} \right\}$$

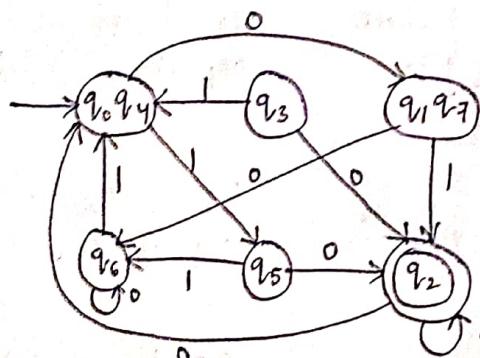
$$\begin{array}{l} q_7 q_1 \\ \delta(q_7, 0) = q_6 \\ \delta(q_1, 0) = q_6 \end{array} \left. \begin{array}{l} \text{Ignore} \\ \text{Ignore} \end{array} \right\}$$

$$\begin{array}{l} \delta(q_7, 1) = q_2 \\ \delta(q_1, 1) = q_2 \end{array} \left. \begin{array}{l} \text{Ignore} \\ \text{Ignore} \end{array} \right\}$$

∴ the unchecked states from table are,

$$(q_3, q_0), (q_7, q_1).$$

$$\therefore (q_0, q_4), (q_1, q_2), (q_3), (q_5), (q_6), (q_7)$$



Moore and Mealy Machines:-

Moore Machine:-

It is a finite state machine where the output depends only on present state.

$\{Q, \Sigma, \Delta, \delta, \lambda, q_0\}$  — 6 tuples in Moore machine

Q — set of states

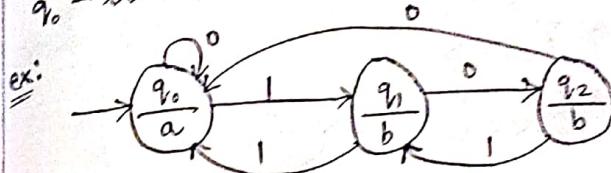
$\Sigma$  — input symbols

$\Delta$  — o/p symbols

$\delta$  — ifp transition function

$\lambda$  — o/p transition function

$q_0$  — initial state



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

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

$$\Delta \rightarrow \{a, b\}$$

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

$$q_0 \xrightarrow[1]{0} q_0$$

$$q_1 \xrightarrow[1]{0} q_1$$

$$q_2 \xrightarrow[1]{0} q_2$$

$$q_0 \xrightarrow[1]{1} q_1$$

$$q_1 \xrightarrow[1]{1} q_2$$

$$q_2 \xrightarrow[1]{1} q_0$$

$$\lambda : Q \rightarrow \Delta$$

$$q_0 \rightarrow a$$

$$q_1 \rightarrow b$$

$$q_2 \rightarrow b$$

$$q_0 \rightarrow q_0 \text{ (initial state)}$$

Transition Table:-

Current state	Next state	O/p
$q_0$	$q_0$	a
$q_1$	$q_2$	b
$q_2$	$q_0$	b

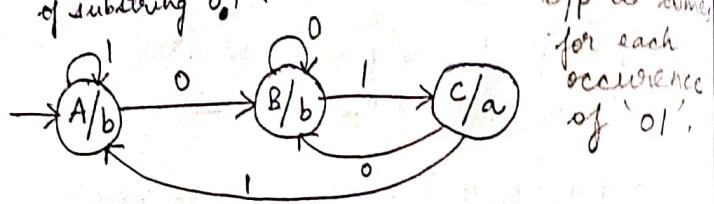
(p.t.o)

→ Suppose 00110 is the input given. What is o/p?  
 00110  
 a a b a a [ irrespective of input first  
 write the o/p of initial state ]

→ 011  
 a a b a

∴  $n \rightarrow n+1$   
 i.e. if there are  $n$  inputs, there will be  
 $n+1$  outputs.

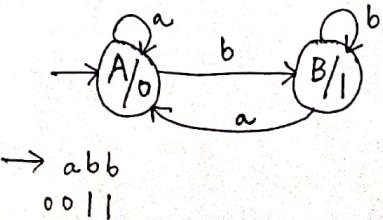
① Construct a Moore machine that takes set of strings over  $\{0,1\}$  and counts the no. of occurrences of substring "01".



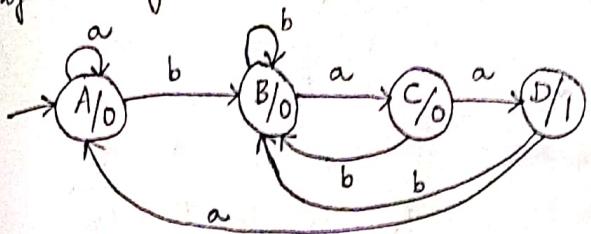
→ Suppose 011001 is i/p given. Then o/p is -  
 b b a b b b a

→ 001  
 b b b a

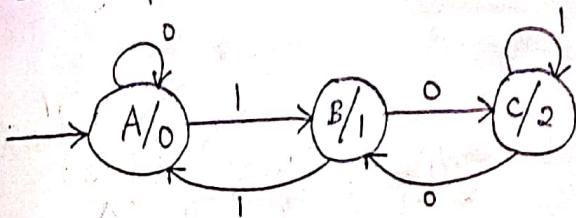
② Construct a Moore Machine that takes set of strings over  $\{a,b\}$  and counts no. of 'b's.



③ Construct a Moore machine that takes set of all strings over  $\{a,b\}$  and counts no. of occurrences of substring 'baa'.

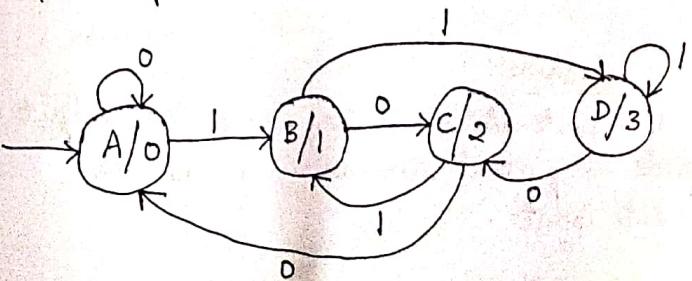


H.W.  
 ① Construct a Moore machine that takes binary numbers as input and produces residue modulo 3 as output.



0 (0)	1 (01)	2 (10)
3 (011)	4 (100)	5 (101)
6 (110)	7 (111)	8 (1000)

② Construct a Moore Machine that takes binary numbers as input and produces residue modulo 4 as output.

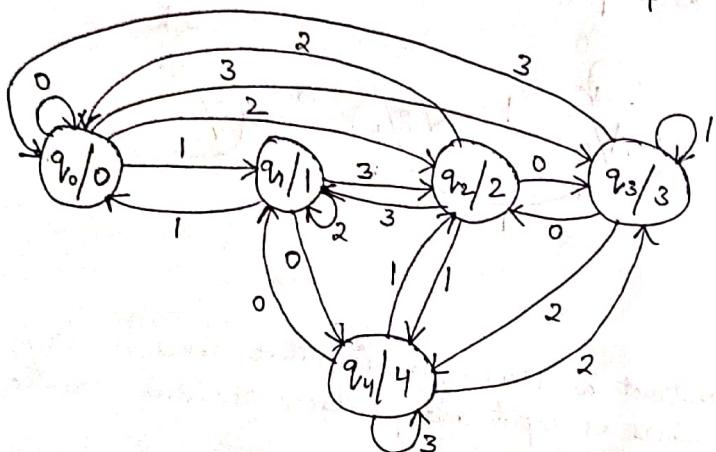


① Construct a Moore Machine that takes gray code numbers as input and produces residue modulo 5 as outputs.

i/p : 0 1 2 3  
o/p : 0 1 2 3 4

Transition table:  
(using trick)

	0	1	2	3
$q_0$	$q_0$	$q_1$	$q_2$	$q_3$
$q_1$	$q_4$	$q_0$	$q_1$	$q_2$
$q_2$	$q_3$	$q_4$	$q_0$	$q_1$
$q_3$	$q_2$	$q_3$	$q_4$	$q_0$
$q_4$	$q_1$	$q_2$	$q_3$	$q_4$



Mealy Machine:-

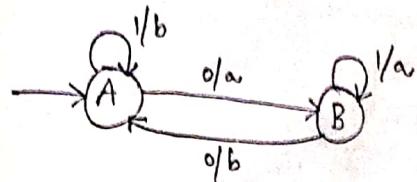
It is a finite state machine where the output depends on present state as well as the present input.

$(Q, \Sigma, \Delta, \delta, \lambda, q_0) \rightarrow$  6 tuples in Mealy Machine

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

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

ex:



$Q \rightarrow \{A, B\}$

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

$\Delta \rightarrow \{a, b\}$

$\delta : A \xrightarrow{0} B$   
 $B \xrightarrow{1} A$

$\lambda : 0 \xrightarrow{A} a$   
 $0 \xrightarrow{B} b$   
1  $\xrightarrow{A} b$   
1  $\xrightarrow{B} a$

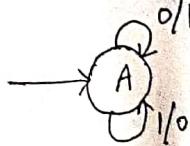
$q_0 : A$

Transition Table:-

Current State (cs)	Next State (ns)	
	i/p = 0	i/p = 1
A	State $\xrightarrow{0} B$	State $\xrightarrow{1} A$
B	State $\xrightarrow{0} A$	State $\xrightarrow{1} B$

① Construct a Mealy machine that produces 1's complement of any binary input strings.

i/p : 0 1  
o/p : 0 1

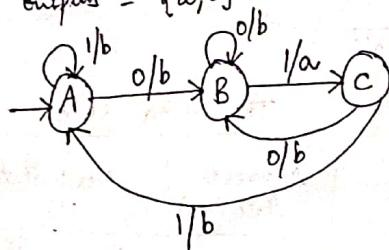


$\therefore n \rightarrow n$   
ie if there are  $n$  inputs there will be  $n$  outputs

- ② Construct a Mealy machine that prints 'a' whenever the sequence '01' is encountered.

$$\text{inputs} = \{0, 1\}$$

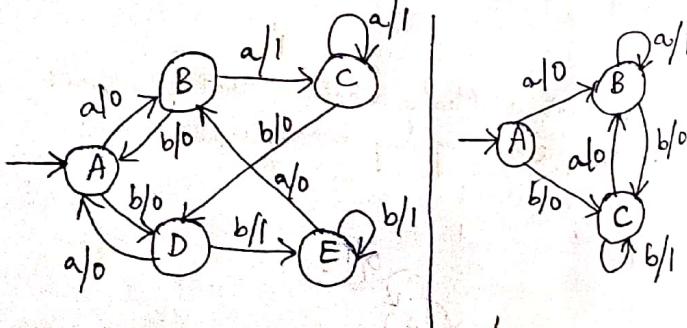
$$\text{outputs} = \{a, b\}$$



→ Suppose 011 as input given  
bab

$$\begin{array}{c|c} \rightarrow 100 \\ \text{bab} & \rightarrow 1001 \\ \hline b & bba \end{array}$$

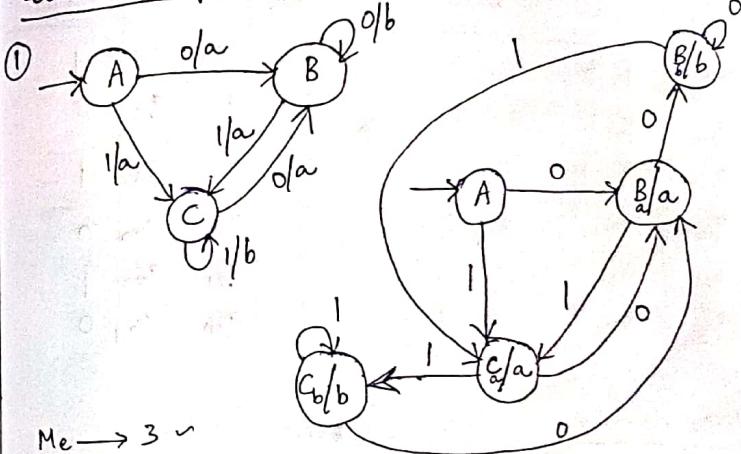
- ③ Design a Mealy machine that accepts 'aa' or 'bb' in the ending of string.



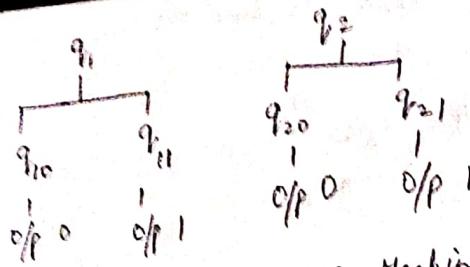
Choose the one with less no. of states.

- ④ Design a Mealy machine to find the 2's complement of a given number.
- |           |
|-----------|
| 001 → 111 |
| 111 → 000 |
| 01 → 11   |
| 11 → 00   |
- Giving input from LSB.

Conversion of Mealy machine to Moore Machine :-



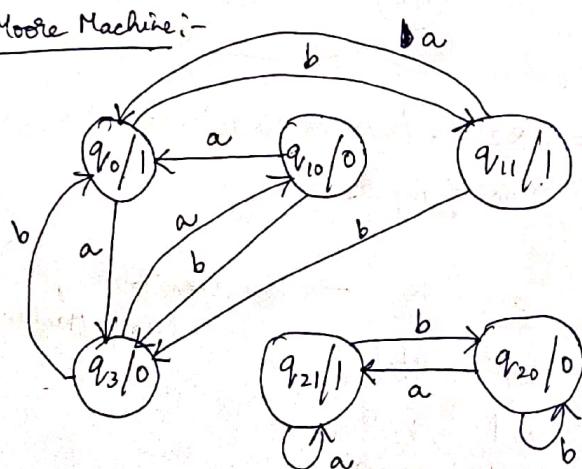
State	Transition Table for Mealy Machine:	
	a	b
$q_{r_0}$	$q_{r_1}, 0$	$q_{r_1}, 1$
$q_{r_1}$	$q_{r_0}, 1$	$q_{r_3}, 0$
$q_{r_2}$	$q_{r_2}, 1$	$q_{r_2}, 0$
$q_{r_3}$	$q_{r_1}, 0$	$q_{r_0}, 1$



Transition table for Moore Machine:-

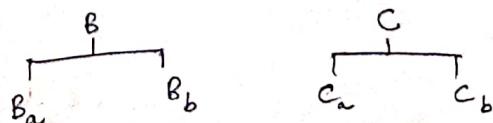
State			State	a	b	O/P
	a	b				
$\rightarrow q_0$	$q_{3,0}$	$q_{11,1}$	$q_0$	$q_{3,0}$	$q_{11,1}$	1
$q_{10}$	$q_{0,1}$	$q_{3,0}$	$q_{10}$	$q_0$	$q_3$	0
$q_{11}$	$q_{0,1}$	$q_{3,0}$	$q_{11}$	$q_0$	$q_3$	1
$q_{20}$	$q_{21,1}$	$q_{20,0}$	$q_{20}$	$q_{21}$	$q_{20}$	0
$q_{21}$	$q_{21,1}$	$q_{20,0}$	$q_{21}$	$q_{21}$	$q_{20}$	1
$q_3$	$q_{10,0}$	$q_{0,1}$	$q_3$	$q_{10}$	$q_0$	0

Moore Machine:-



Q(1) Transition table for Mealy Machine:-

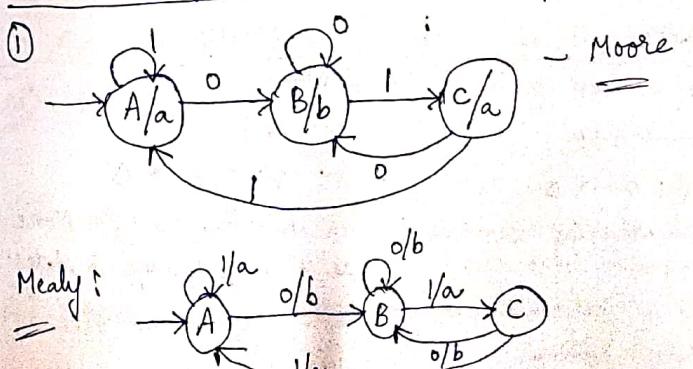
State	0		1	
	B, a	C, a	B, b	C, b
A				
B				
C				



Transition Table for Moore Machine :-

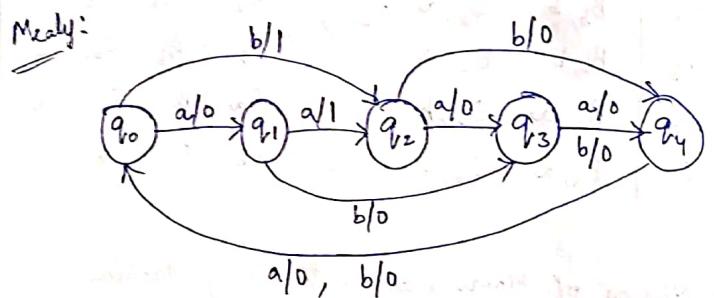
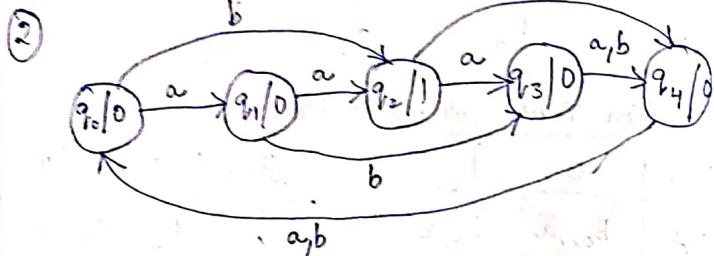
State	0		1		State	0		1		O/P
	0	1	0	1	A	B <sub>a</sub>	C <sub>a</sub>	B <sub>b</sub>	C <sub>b</sub>	
$\rightarrow A$	$B_a, a$	$C_a, a$			A	$B_a$	$C_a$			-
$B_a$	$B_b, b$	$C_a, a$			$B_a$	$B_b$	$C_a$	$a$		
$B_b$	$B_b, b$	$C_a, a$			$B_b$	$B_b$	$C_a$	$b$		
$C_a$	$B_a, a$	$C_b, b$			$C_a$	$B_a$	$C_b$	$a$		
$C_b$	$B_a, a$	$C_b, b$			$C_b$	$B_a$	$C_b$	$b$		

Conversion of Moore machine to Mealy Machine :-



Moore Transition Table :-		
State	0	1
$\rightarrow A$	B	A
B	C	b
C	A	a

State	0		1	
	NS	O/P	NS	O/P
$\rightarrow A$	B	b	A	a
	B	b	C	a
	C	b	A	a



- |   |   |
|---|---|
| 1. Definition.  | 1. Definition.  |
| 2. $n \rightarrow n+1$  | 2. $n \rightarrow n$  |
| 3. $\lambda : Q \rightarrow \Delta$   | 3. $\lambda : Q \times \Sigma \rightarrow \Delta$                                   |
| 4. When converting Moore $\rightarrow$ Mealy<br>No change in no. of states. | 4. When converting Mealy $\rightarrow$ Moore<br>there is increase/change in states. |
| 5. O/p is synchronous with<br>clock. (Slower).                              | 5. O/p is asynchronous.<br>(Faster).  |

Mealy Transition Table:-

31/5/21

Unit - 2.

Q1: Write down the regular expression where  
 $\Sigma = a, b$ , having,

- (i) no string      (ii) length 0      (iii) length 1  
 $\hookrightarrow RE - \emptyset$        $\hookrightarrow \epsilon$        $\hookrightarrow (a+b)$

(iv) length 2      (v) length 3  
 $\hookrightarrow aa + ab + ba + bb$        $\hookrightarrow (a+b)(a+b)(a+b)$   
 $a(a+b) + b(a+b)$

$$\begin{aligned} &= (a+b)(a+b) \\ &\downarrow \\ &(E+a+b) \quad \downarrow \\ &R.E \quad \text{Regular set} \end{aligned}$$

② Write the regular expression for all strings of 0's and 1's.

Sol<sup>n</sup>: First write the language for 0's and 1's.  
 $\{ \epsilon, 0, 1, 01, 00, 10, 11, 000, \dots \}$

Regular expression is  $(0+1)^*$ .

③ Set of all strings of 0's and 1's ending with 00.

Sol<sup>n</sup>: Language  $\{ 00, 000, 100, 1000, 0100, 01000, \dots \}$   
 $\{ \epsilon, 0, 1, 10, 01, 11, 00, \dots \}$

Regular expression is  $(0+1)^*00$ .

④ Set of all strings of 0's and 1's beginning with 0 and ending with 1.

Exn: Language  $\{01, 001, 011, 0001, 0101, \dots\}$   
 $\{ \epsilon, 0, 1, 00, 10, 11, 01, \dots \}$

R.E is  $(0+1)^*$

⑤ Set of all strings with atleast 2 consecutive 0's.

Exn: Language  $\{00, 001, 100, 000, \dots\}$

R.E is  $(0+1)^* 00 (0+1)^*$

⑥  $\Sigma = a, b$ , write R.E where the length of strings is atleast 2.

Exn: Language  $\{bb, ba, bab, babb, baaba, \dots\}$

R.E is  $(a+b)(a+b)(a+b)^*$

⑦ Write R.E where  $\Sigma = a, b$ , having length atleast 2.

Exn:  $L = \{\epsilon, a, b, aa, bb, ab, ba\}$

R.E is  $(\epsilon+a+b)(\epsilon+a+b)$

⑧ Write R.E where  $\Sigma = a, b$  having length of string is even.

Exn:  $L = \{\epsilon, aaa, ab, abab, ababb, \dots\}$

R.E is  $((a+b)(a+b))^*$  → if we put 0 in place of  $a$  then length 0.

⑨ Write R.E where  $\Sigma = a, b$  having length of string is odd.

Exn:  $L = \{a, b, aaa, bbb, \dots\}$

R.E is  $(a+b)/((a+b)(a+b))^*$

⑩ Write R.E where  $\Sigma = a, b$  and whose length is divisible by 3.

Exn:  $L = \{\epsilon, aaa, bbb, aabbab, \dots\}$

R.E is  $((a+b)(a+b)(a+b))^*$

### Identity Rules for Regular Expressions :-

If P and Q are two equivalent R.Es (ie P and Q represent the same set of strings), then to simplify the R.Es, the foll. identity rules can be used:

1)  $\phi + R = R, \epsilon + R = R + \epsilon$

2)  $\epsilon R = R \epsilon = R$

3)  $R + R = R$

4)  $RR^* = R^* R = R^+$

5)  $\epsilon + RR^* = R^* = \epsilon + R^* R$

6)  $\phi R = R \phi = \phi$

7)  $\epsilon^* = \epsilon, \phi^* = \epsilon$

8)  $(R^*)^* = R^*$

9)  $(PQ)^* P = P(QP)^*$

10)  $(P+Q)R = PR + QR$

11)  $R(P+Q) = RP + RQ$

12)  $(P+Q)^* = (P^*Q^*)^* = (P^* + Q^*)^*$

Q1 Write down the R.E whose length is congruent to  $2 \pmod{3}$ .  
 $\hookrightarrow$  any number divided by 3 must give remainder 2.

Soln: Lengths will be 2, 5, 8, 11, ...  
 R.E is  $(a+b)(a+b) \cdot ((a+b)(a+b)(a+b))^*$

$\star = 0$  len = 2  
 $\star = 1$  len = 5  
 $\star = 2$  len = 8

Q2 R.E for length congruent to  $3 \pmod{4}$ .

Soln: Lengths will be 3, 7, 11, 15, ...  
 R.E is  $(a+b)(a+b)(a+b)/((a+b)(a+b)(a+b)(a+b))^*$

Q3 Write R.E where  $\Sigma = a, b$  whose no. of a's are exactly 2.

Soln:  $L = \{aa, aab, baab, aabb, abaa, \dots\}$   
 R.E is  $b^* a b^* a b^*$

Q4 Write R.E whose having atleast 2 a's.

Soln:  $L = \{aa, abaa, baa, ababa, \dots\}$   
 R.E is  $b^* a b^* a (a+b)^*$

Q5 Write R.E having almost 2 a's.

Soln:  $L = \{ab, bb, aab, a, abab, \dots\}$   
 R.E is  $b^*(\epsilon+a)b^*(\epsilon+a)b^*$

$$\begin{aligned} \epsilon \epsilon &\rightarrow 0 \text{ a's} \\ \epsilon a &\rightarrow 1 \text{ a} \\ a a &\rightarrow 2 \text{ a's} \end{aligned}$$

Q6 Write R.E whose no. of a's is even.  
 Soln:  $L = \{b, aab, baab, babaaba, \dots\}$   
 R.E is  $b^* + (b^* a b^* a b^*)^*$

Q7 Write R.E for set of all strings beginning with 1 or 0 but not having consecutive 0's.

Soln:  $L = \{10, 101, 010, 0101, \dots\}$   
 R.E for string that does not contain 2 consecutive 0's is  $(1+10)^*$

R.E starting with 0 or 1 is  $(0+1)$

Final R.E is  $(0+\epsilon)(1+10)^*$  [ $\because$  for 10 to be acceptable].

$$\epsilon 10 = 10$$

Q8 Write R.E whose set of strings is 0's followed by 1, followed by 2 such that atleast one 0 followed by atleast one 1  $\hookrightarrow$  which is followed by atleast one 2.

Soln: R.E is  $0^+ 1^+ 2^+$  ( $\because$  we cannot put \* since atleast 012 has to be there ( $\epsilon$  is not accepted)).

Q9 Write a R.E where  $\Sigma = 0, 1$  whose last 2 symbols are same.

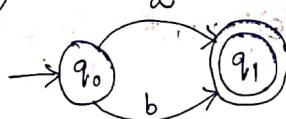
Soln:  $L = \{00, 11, 000, 011, 100, \dots\}$   
 R.E is  $(0+1)^*(00+11)$

Q10 Write down the R.E in which every 0 is

followed by atleast two 1's  
 soln:  $L = \{1011, 011, 011011, \dots\}$   
 R.E is  $(1+011)^*$   
 $\because (0+1)^*$  contains all strings of 0 and 1's with no condition.  
 but in this Q 0 must be followed by 11  
 so  $(011+1)^*$

Regular expression to Finite Automata :-

①  $(a+b)$



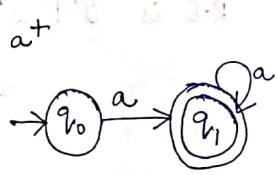
②  $(ab)$



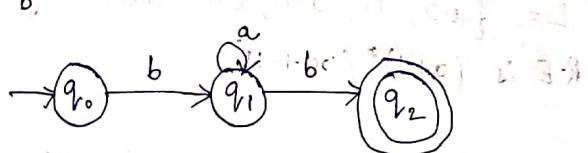
③  $a^*$



④  $a^+$



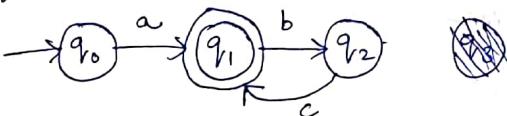
⑤  $ba^*b$



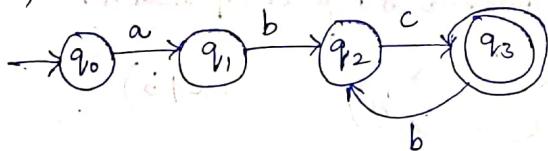
⑥  $(a+b)c$



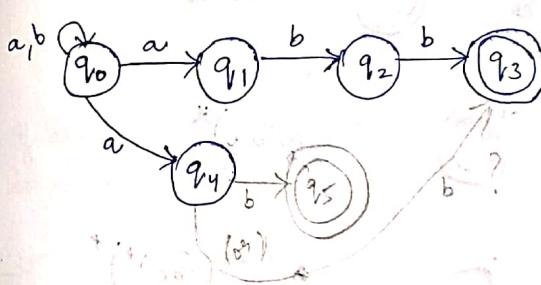
⑦  $a(bc)^*$



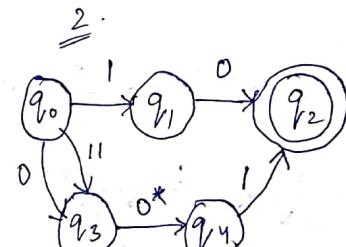
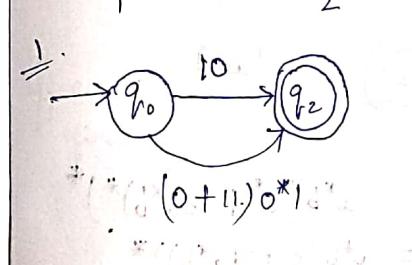
⑧  $a(bc)^{++}$

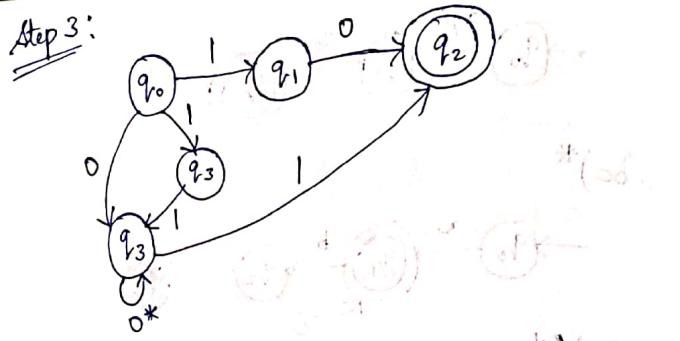


⑨  $(a+b)^*(abb + a^+b)$



⑩  $10 + (0+11)0^{*}1$





Finite Automata to Regular Expression :-

- ①  $(a+b)$
- ②  $(ab^*)^*(dc)$
- ③  $a(ba)^*$
- ④  $\bullet (a^* + (bc)^*)^*$   
 $= (a+bc)^* = (a^*(bc))^*$
- ⑤  $(ab)^*$
- ⑥  $a^*b(c^* + (da^*b)^*)^*$   
 $= a^*b(c + (da^*b))^*$

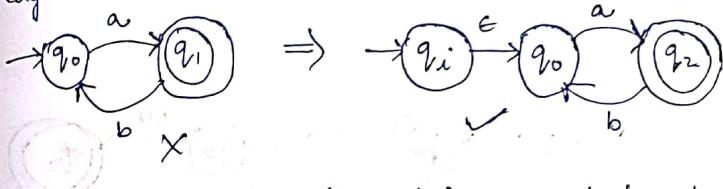
Methods to convert Finite Automata to R.E :-

1. State Elimination method :-

2. Arden's theorem.

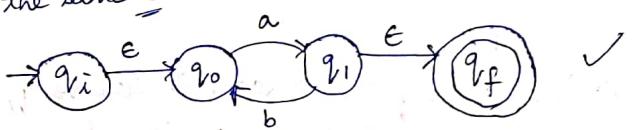
State Elimination method :-

1) Initial state should not contain any incoming edges.

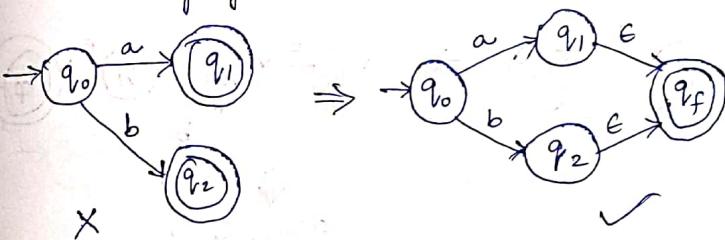


2) Final state should not contain any outgoing edge.

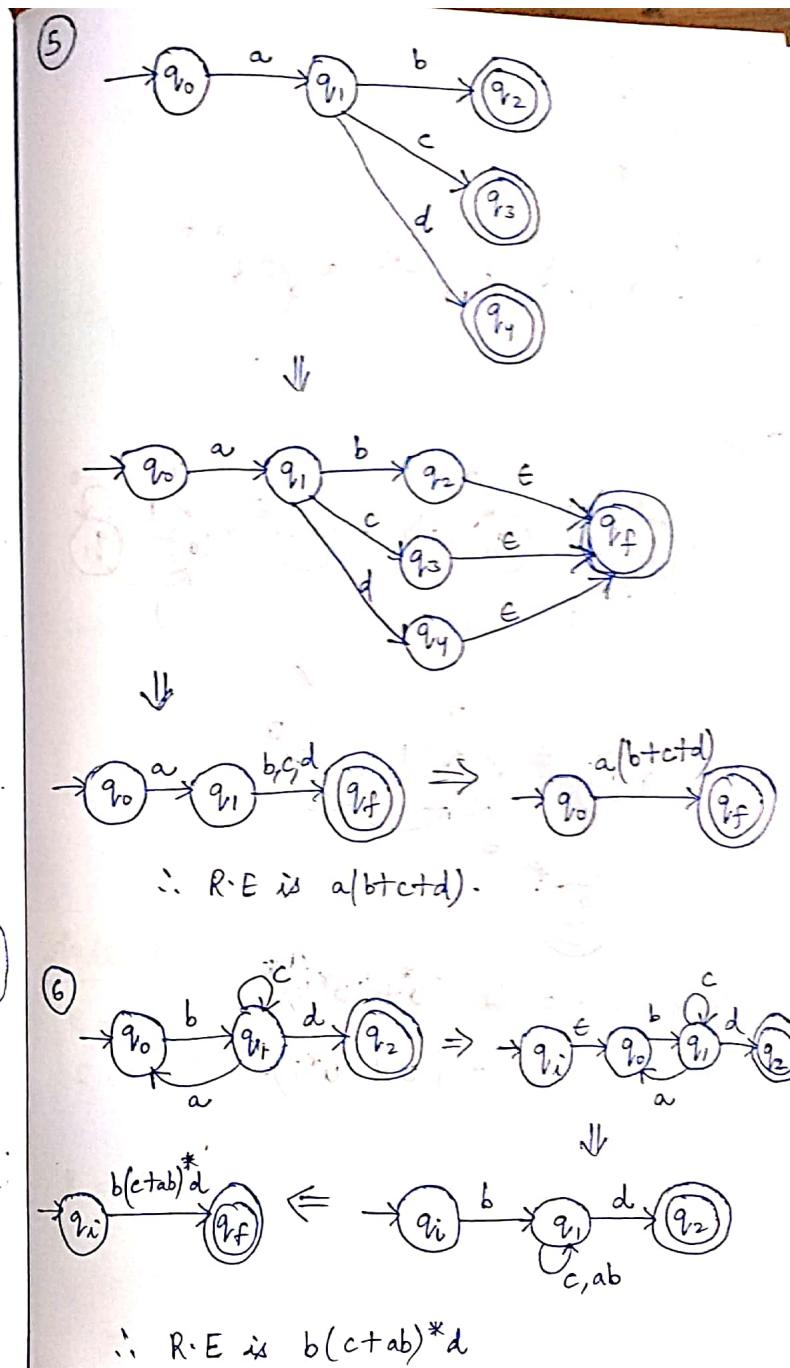
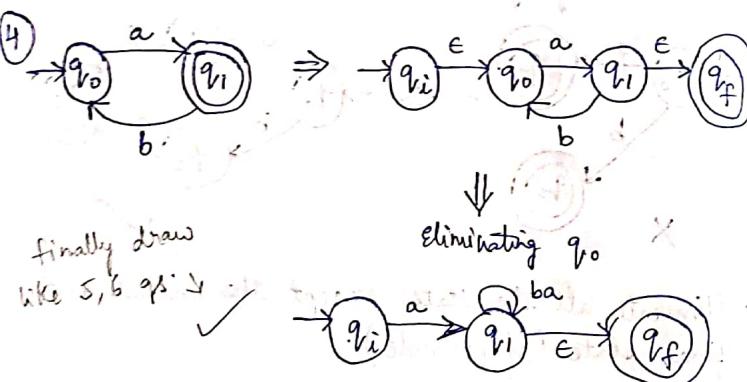
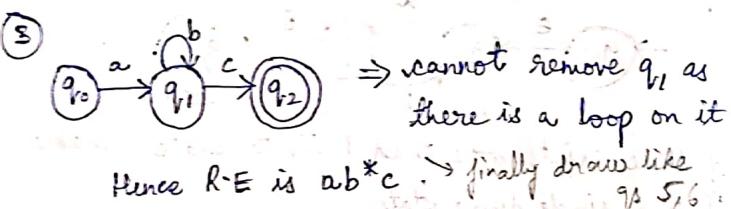
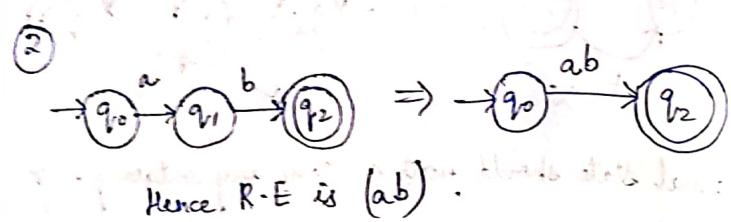
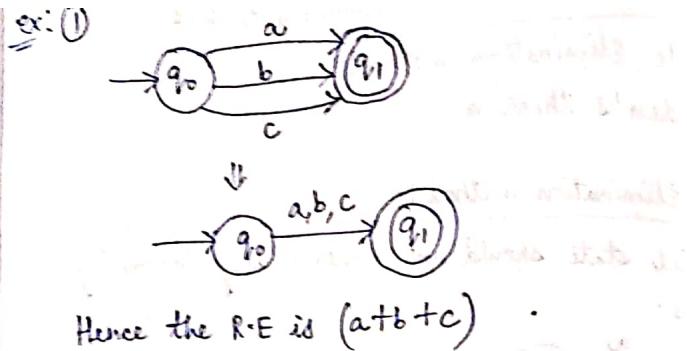
→ for the same ex:

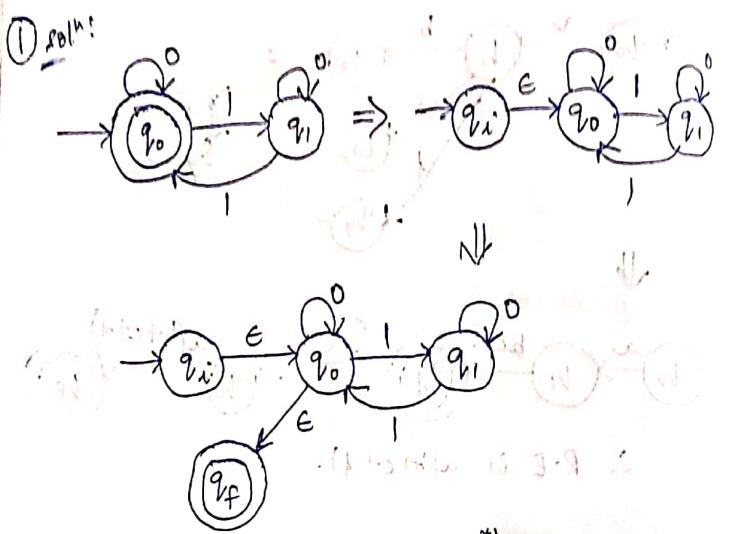
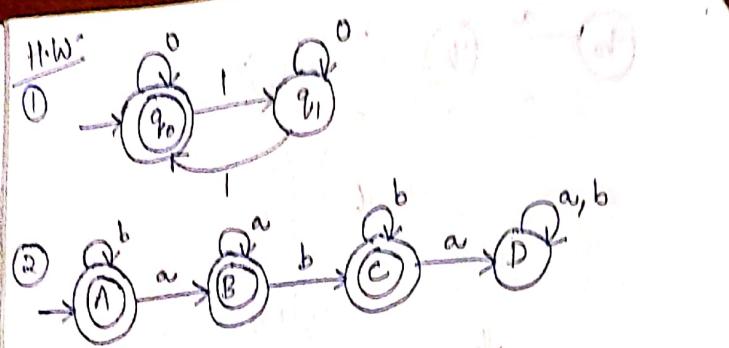


3) If there exists more than 1 finite state, make it a single finite state.



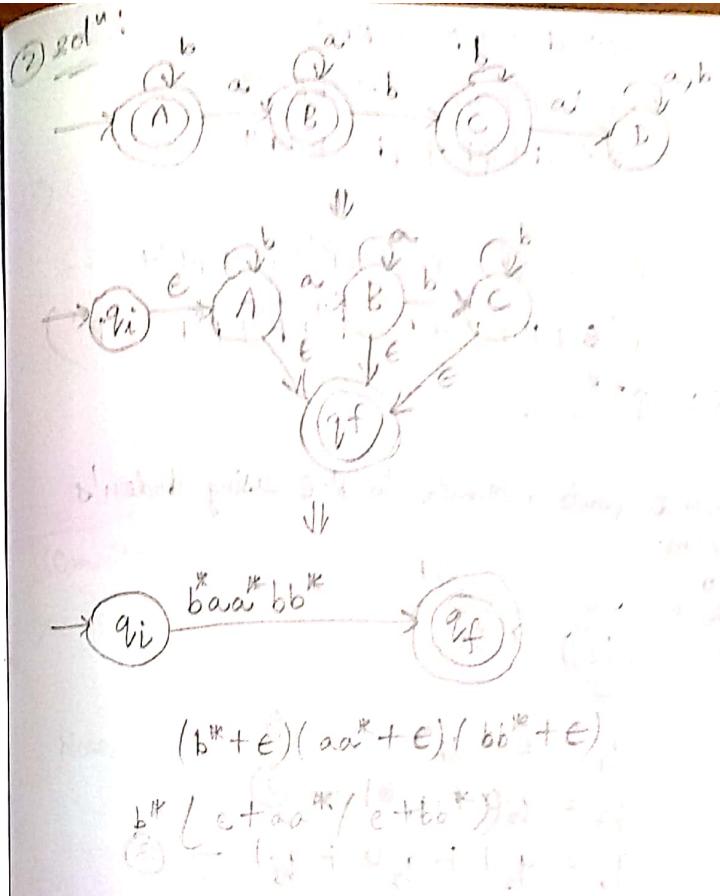
4) Eliminate all the states except the initial and final state (if possible).





Hence the R.E is  $(0^* + (10^*)^*)^*$

$$= (0 + 10^*)^*$$



Auden's Theorem:

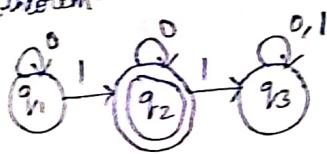
$R = Q + RP$  (P does not belong to null)  
 $R = QP^*$

$$\begin{aligned}
 R &= Q + RP \\
 &= Q + (Q + RP)P \\
 &= Q + QP + RP^2 \\
 &= Q + QP + (Q + RP)P^2
 \end{aligned}
 \quad \xrightarrow{\text{Proof:}}$$

$$\begin{aligned}
 &= Q + QP + QP^2 + RP^3 \\
 &= Q + QP + QP^2 + (Q + RP)P^3 \\
 &= Q + QP + QP^2 + QP^3 + RP^4 \\
 &\vdots \\
 &= Q + QP + QP^2 + QP^3 + QP^4 + QP^5 + QP^6 + \dots \\
 &= Q(E + P + P^2 + P^3 + P^4 + P^5 + P^6 + \dots P^n)
 \end{aligned}$$

$$\therefore R = QP^*$$

① Convert finite automata to R-E using Arden's theorem.



$$q_1 = \epsilon + q_1 0 \quad \text{--- } ①$$

$$q_2 = q_2 0 + q_1 1 \quad \text{--- } ②$$

$$q_3 = q_2 1 + q_3 0 + q_1 1 \quad \text{--- } ③$$

$$\text{Consider eq. } ① \Rightarrow q_1 = \frac{\epsilon + q_1 0}{R} = \frac{\epsilon + q_1 0}{Q + RP}$$

$$q_1 = \epsilon \cdot 0^* = 0^* \quad \text{--- } ④$$

$$\text{Now, } q_2 = q_2 0 + q_1 0$$

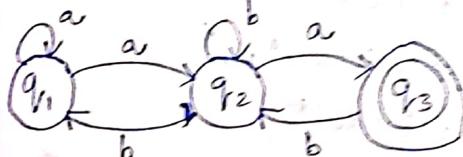
$$\frac{q_2}{R} = \frac{q_2 0}{R} + \frac{q_1 0}{P} \quad \text{--- } ⑤$$

$$q_2 = 0^* 10^*$$

$q_3$  is a dead state. So delete it.

$\therefore R\text{-E is } 0^* 10^*$

②



$$q_1 = \epsilon + q_1 a + q_2 b \quad \text{--- } ①$$

$$q_2 = q_2 b + q_1 a + q_3 b \quad \text{--- } ②$$

$$q_3 = q_2 a \quad \text{--- } ③$$

$$\text{Consider eq. } ①, \quad \frac{q_1}{R} = \frac{\epsilon + q_1 a + q_2 b}{Q} = \frac{\epsilon + q_2 b}{R} = \frac{q_2 b}{R P}$$

$$q_1 = (\epsilon + q_2 b) a^* = q_2 b a^*$$

$$\text{Now, } q_2 = \frac{q_2 b + q_1 a + q_3 b}{R} = \frac{q_2 b}{R P}$$

$$q_2 = (q_1 a + q_3 b) b^* \quad \times$$

put ③ in ②,

$$q_2 = q_2 b + q_1 a + q_2 ab$$

$$\frac{q_2}{R} = \frac{q_2 b}{Q} + \frac{q_1 a}{R P}$$

$$q_2 = q_1 a (b + ab)^* \quad \text{--- } ④$$

put ④ in ①,

$$q_1 = \epsilon + q_1 a + q_1 a (b + ab)^* b$$

$$q_1 = \frac{e + q_1 | a + a(b+ab)^* b}{R} \quad \text{eq. 1}$$

$$= \frac{e}{R} + \frac{q_1 | a(b+ab)^* b}{R} \quad \text{P.C.T.O. N 3.4}$$

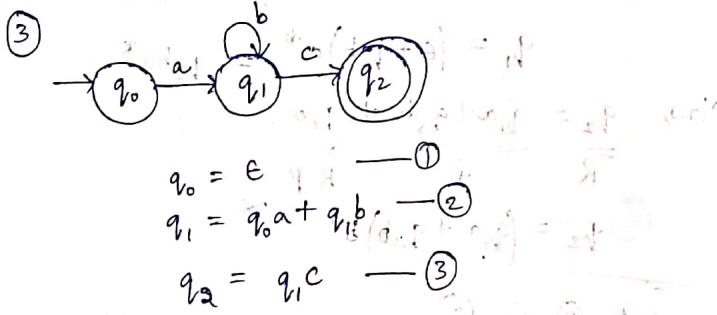
$$q_1 = e(a + a(b+ab)^* b)^*$$

$$= (a + a(b+ab)^* b)^*$$

$$q_2 = q_1 a(b+ab)^* \\ = (a + a(b+ab)^* b)^* a(b+ab)^*$$

$$q_3 = q_2 a \\ = a(a + a(b+ab)^* b)^* a(b+ab)^*$$

$$q_3 = a^2(b+ab)^*(a + a(b+ab)^* b)^* \\ \downarrow \text{is R.E.}$$



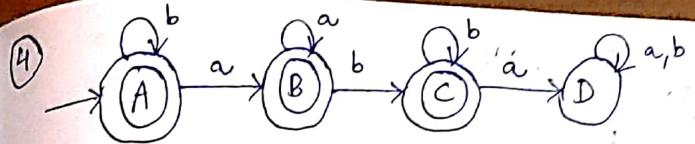
Now,  $\frac{q_1}{R} = \frac{a + q_1 b}{R P}$

$$q_1 = ab^*$$

$$q_2 = q_1 c$$

$$q_2 = ab^* c$$

$\therefore$  The R.E. is  $ab^* c$ .



$$A = Ab + E \quad \text{--- ①}$$

$$B = Aa + Ba \quad \text{--- ②}$$

$$C = Bb + Eb \quad \text{--- ③}$$

$$\text{eq. ①} \Rightarrow \frac{A}{R} = \frac{Ab + E}{R P} \quad \text{--- ④}$$

$$A = Eb^* \Rightarrow A = b^* \quad \text{--- ④}$$

$$\text{eq. ②} \Rightarrow B = Aa + Ba$$

$$\frac{B}{R} = \frac{b^* a + Ba}{R P}$$

$$B = b^* aa^* \quad \text{--- ⑤}$$

$$\text{eq. ③} \Rightarrow C = Bb + Eb$$

$$\frac{C}{R} = \frac{b^* aa^* b + Eb}{R P}$$

$$C = b^* aa^* bb^* \quad \text{--- ⑥}$$

Final R.E is  $A+B+C$  [ $\because A, B, C$  are final states].

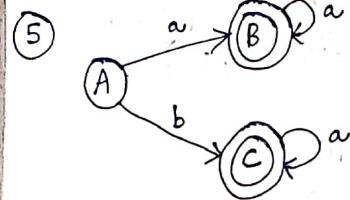
$$= b^* + b^* aa^* + b^* aa^* bb^*$$

$$= b^*(aa^* + E + aa^* bb^*)$$

$$= b^*(aa^*(bb^* + E) + E)$$

$$= b^*(aa^* b^* + E)$$

$$\therefore bb^* + E = b^*$$



$$A = \epsilon$$

$$B = Aa + Ba$$

$$C = Ab + Ca$$

$$\text{eq. } (2) \Rightarrow B = Ea + Ba$$

$$B = a + Ba$$

$$R \stackrel{?}{=} \overline{Q} \overline{R} P$$

$$B = aa^*$$

$$\text{eq. } (3) \Rightarrow C = Ab + Ca$$

$$= Eb + Ca$$

$$R \stackrel{?}{=} \overline{Q} \overline{R} P$$

$$C = ba^* \quad (5)$$

$$\text{Final R.E. is } B + C$$

$$= aa^* + ba^*$$

\* Finite languages  $\rightarrow$  is accepted by finite automata  
 $\rightarrow$  if it is regular language.

\* Infinite language  $\rightarrow$  cannot be accepted if it is not regular.

Ex:  $a^n b^m / n, m \geq 1$

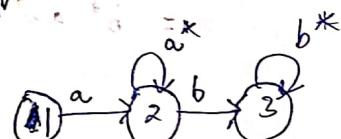
ab

aabb

aab

abb

Hence it is Regular language.



no memory required, because

$n, m$  are independent of each other.

Ex:  $a^*$  is finite language because we can obtain a finite automata.

(1)  $a^n b^n / n \leq 100$

$$a^n b^n / n \geq 1$$

$$2 \rightarrow aabb$$

$$3 \rightarrow aaabbb$$

Here no. of a's & b's are same and infinite.

So it is not regular language.

(2)  $a^n b^m c^k / n, m, k \geq 1$

after taking  $n$  a's, the b does not depend on  $n$ . a's & same is for c. So no memory required.

$\therefore$  It is regular language.

(3)  $ww^R / |w| = 2, \Sigma = \{a, b\}$

Given that length of  $w$  is 2.

if  $w = aa \rightarrow aaaa$

if  $w = ab \rightarrow abba$

if  $w = bb \rightarrow bbbb$

if  $w = ba \rightarrow baab$

$w^R$  is

reverse of  $w$

$\therefore$  It is regular language.

(4)  $ww / w \in (a, b)^*$

if  $w = abb \rightarrow abbabb$

if  $w = aa \rightarrow aaaa$

the length can be infinite.

So it is not regular language.

$$(5) \underline{a^i b^{2j}} \mid i, j \geq 1$$

$\begin{array}{c|c} i & j \\ \hline 1 & 1 \\ 1 & 2 \end{array} \rightarrow \begin{array}{l} abb \\ abbbb \end{array}$

This is also infinite, but the no. of b's do not depend on a.  
 $\therefore$  The given language is regular language.

$$(6) a^n \mid n \text{ is even}$$

$$n=0 \rightarrow a^0$$

$$n=2 \rightarrow aaaa$$

$$n=4 \rightarrow aaaaa$$

$$\text{A} \quad \{a, a^2, a^4, \dots\}$$

$\therefore$  Regular.

( $\because$  we have finite automata, also we have a pattern).

$$(8) a^n \mid n \text{ is prime}$$

$$\{a^2, a^3, a^5, a^7, a^11, a^13, \dots\}$$

No pattern.

$\therefore$  Not Regular language.

$$(10) a^{2^n} \mid n \geq 1$$

$$\{a^2, a^4, a^8, a^{16}, \dots\}$$

We can't construct any finite automata using this pattern.

$\therefore$  It is not R.L.

$$(7) a^n \mid n \text{ is odd}$$

$$n=1 \rightarrow a$$

$$n=3 \rightarrow aaa$$

$$n=5 \rightarrow aaaaa$$

$$\{a, a^3, a^5, a^7, \dots\}$$

Regular.

$aabbbaabbb$

$$(11) a^n b^{n+m} c^m \mid n, m \geq 1$$

The no. of b's are dependent on no. of a's and no. of c's.  
 And it can be infinite and can't be stored in memory.  
 $\therefore$  It is not R.L.

\* (If there is dependence, then not R.L.)

$$(12) a^n b^{n+m} \mid n, m \leq 100000$$

Here n, m are finite - So finite memory is only required.

$\therefore$  The given lang. is R.L.

Pumping Lemma :-

If L is an infinite lang., then there exists some integer 'n' such that any string  $w \in L$  has length greater than equal to 'n' i.e.,

$|w| \geq n$  then we can divide into 3 parts  $w = xyz$  satisfy the foll. conditions

i) for each  $i \geq 0$ ,  $xyz^i \in L$ .

ii)  $|y| > 0$

iii)  $|xy| \leq n$ .

If a given lang. passes the Pumping Lemma conditions then we can't decide (R.L./N.R.L.) but if the conditions fail then lang. is not R.L.

① Prove the lang.  $a^n b^{2n} \mid n \geq 0$   
 by using P.L.  $\rightarrow$  if L is R.L. then P.L. is also R.L.  
 for  $n=3 \rightarrow \underline{aaabbbbbb}$   $\underline{xyz}$  (can divide  
 in any manner)  
 $xy^2z \rightarrow \underline{xy^2z}$  (making  $i=2$ ) (such that condition  
 satisfy)  
 $\hookrightarrow \underline{aaabbbbbb} \underline{aaabbbbbb} \notin L \quad [ \because a^n \neq b^{2n} ]$   
 $\therefore \text{Not R.L.}$

②  $a^n b^n \mid n >= 1$   
 $\rightarrow a^n b^n \quad n=5 \text{ (say)} \quad |xy| \leq n$   
 $\underline{aaaaa} \underline{bbbb} \quad 5 \leq 5 \checkmark$   
 $xy^2z \rightarrow \underline{xy^2z}$  (taking  $i=2$ )  
 $\underline{aaa} \underline{aa} \underline{aa} \underline{bbb} \underline{bb} \notin L \quad [ a^n \neq b^n ]$   
 $\therefore \text{Not R.L.}$

③  $a^n \mid n \geq 1$   
 $\rightarrow a^n, n=5 \text{ (say)}$   
 $\underline{aaaaa} \underline{x} \underline{y} \underline{z}$   
 $xy^2z \rightarrow \underline{xy^2z}$  (taking  $i=2$ )  
 $\underline{aaa} \underline{aa} \underline{aa} \in L$

P.L conditions passed. So cannot decide R.L or not R.L.

④  $L = WW \quad , w \text{ belongs to } (a,b)^*$   
 if  $w=aba \rightarrow \underline{aba} \underline{aba}$   
 $xy^iz \rightarrow \underline{xy^i z}$  (taking  $i=4$ ).  
 $abbbbaaba \notin L$   
 $\therefore \text{Not R.L.}$

### Closure Properties of Regular sets :- ( $L \& M$ are 2 R.L.)

- Union of two regular languages is regular. ( $L \cup M$ )
- Intersection of two regular languages is regular ( $L \cap M$ )
- Complement of a language is regular. ( $\Sigma^* - L$ )
- Difference of two regular languages is regular. ( $L - M$ )
- Reversal of a R.L is regular.  
 ex: if  $L = \{001, 110\}$ , then  $L(R) = \{100, 011\}$
- Closure of a R.L is regular ( $L^*$ ).
- Concatenation of R.L's is regular. ( $L M$ ).
- Homomorphism of a R.L is regular.  
 A homomorphism is the substitution of strings for symbols.

$$h(L) = \{ h(w) \mid w \in L \}$$

$$h: \Sigma \text{ to } \Psi^*$$

$$\text{ex: } \Sigma = \{0,1\}, \Psi = \{a,b\}$$

$$\text{Let } h(0) = aa$$

$$h(1) = bb$$

$$\text{If } L = \{00, 11\}$$

$$\text{then } h(L) = \{aaaa, bbbb\}$$

### Inverse Homomorphism :-

$$\begin{aligned} h(0) &= a \\ h(1) &= b \\ h(2) &= ab \\ h(L) &= \{abab\} \end{aligned}$$

where  $\Sigma = \{0, 1\}$   
 $\gamma = \{a, b\}$ .

$$\Rightarrow h^{-1}(L) = \left\{ \frac{0101}{abab}, \frac{22}{abab}, \frac{012}{abab}, \frac{201}{abab} \right\}$$

$$\text{So, } h(h^{-1}(L)) = \{abab, abab, abab, abab\}$$

$$= \{abab\}$$

If the lang. is regular, then inverse of homomorphism is also regular.

### Applications of R.L :-

- ① Lexical Analysis.
- ② Finding patterns.

$$(a^2b)^* = (ab)^*$$

$$(ab)^* = \{a, b\}^*$$

### Unit - 3

RG  
 $\downarrow$   
 RL  
 $\downarrow$   
 FA

A grammar is regular if it has rules of form  $A \rightarrow a$  (or)  $A \rightarrow Ba$  (or)  $A \rightarrow \epsilon$

16/6/21:

Unit - 3

### Grammar:-

Grammar is a standard way of representing lang.

$$G_1 = \{V, T, P, S\} \rightarrow 4 \text{ tuples.}$$

V - Variables (upper case)

T - terminals (lower case)

P - Production rule

S - Start symbol

(or)

$$S \rightarrow aSb \mid \epsilon \rightarrow \text{production}$$

a, b : terminals, S : variable

Consider  $S \rightarrow aSb$  (S on replacing with  $aSb$ )

$$a\underline{aSb}b \rightarrow aaa\underline{Sbb}b$$

if  $S = \epsilon$  then  $aabb \quad aaabb$

Hence the lang. generated by given grammar is  $a^n b^n$

$$\begin{array}{l} ① \quad S \rightarrow SS \\ S \rightarrow aSb \\ S \rightarrow bSa \\ S \rightarrow \epsilon \end{array} \Rightarrow S \rightarrow SS \mid aSb \mid bSa \mid \epsilon$$

soln:  $S \rightarrow SS$   
 $aSb \ aSb \rightarrow abab$   
 $bSa \ bSa \rightarrow abab abab$

$$S \rightarrow SS \qquad \qquad \qquad S \rightarrow SS \\ aSb \ bSa \rightarrow abba \qquad aSb \epsilon \rightarrow ab$$

$$S \rightarrow SS \qquad \qquad \qquad S \rightarrow SS \\ bSa \ bSa \rightarrow babab \qquad bSa \epsilon \rightarrow ab$$

$\therefore$  The lang. is  $(ab + ba)^*$

Regular Grammar (RG) :-

Left Linear Grammar  
(LLG)

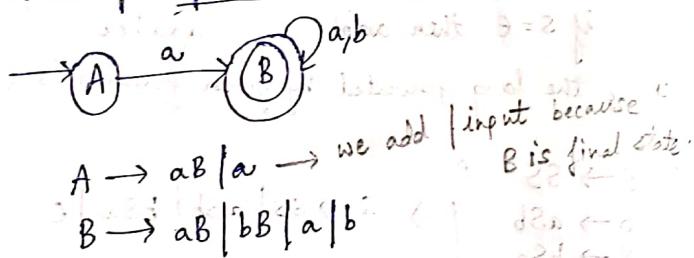
$$A \rightarrow a$$

$$A \rightarrow Ba$$

$$A \rightarrow a | Ba$$

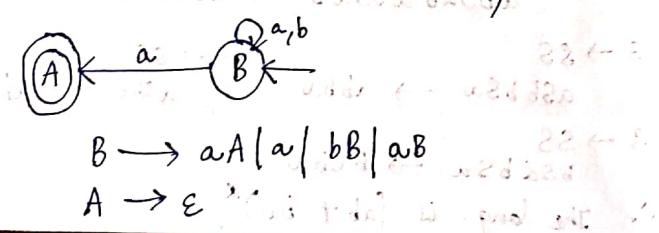
ex:  $S \rightarrow abb$   
is not a R.G because it is neither LLG nor RLG.

① Convert given finite automata to R.G.



Now convert this into LLG.

Step 1: Reverse the finite automata (i.e. initial & final states are exchanged & arrow directions are also reversed).



Right Linear Grammar  
(RLG)

$$A \rightarrow a$$

$$A \rightarrow ab$$

$$A \rightarrow a | ab$$

Step 2: Now just change as follows.

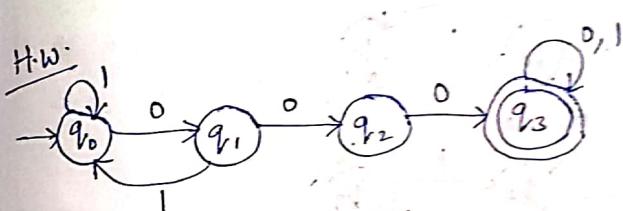
$$B \rightarrow Aa | a | Bb | Ba$$

$$A \rightarrow \epsilon$$

Hence this is the LLG.

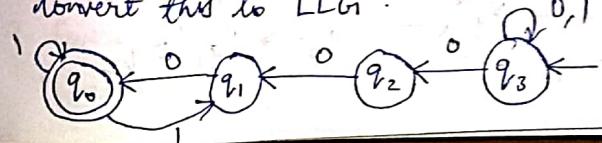
Rules to convert finite automata to R.G :-

- \* If there is a transition of the form  $\delta(q, a) \rightarrow p$ , then add as production " $q \rightarrow ap$ " if  $p$  is non-final state.
- \* If there is a transition of the form  $\delta(q, a) \rightarrow p$ , then add the production " $q \rightarrow ap | a$ " if  $p$  is final state.



$$\left. \begin{aligned} q_0 &\rightarrow | q_0 | 0 q_1 \\ q_1 &\rightarrow | q_0 | 0 q_2 \\ q_2 &\rightarrow 0 q_3 | 0 \\ q_3 &\rightarrow 0 q_3 | 0. 1 q_3 | 0 | \end{aligned} \right\} \text{RLG.}$$

Convert this to LLG.

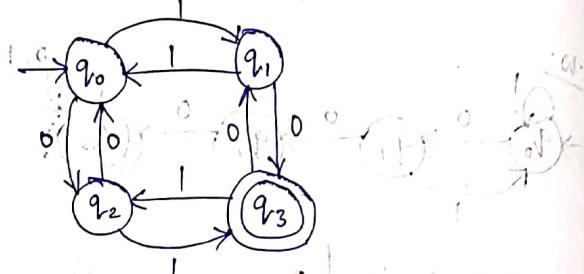


$$\begin{aligned}
 q_3 &\rightarrow 0q_3 \mid 1q_3 \mid 0q_2 \\
 q_2 &\rightarrow 0q_1 \\
 q_1 &\rightarrow 0q_0 \mid 0 \\
 q_0 &\rightarrow 1q_0 \mid 1 \mid 1q_1
 \end{aligned}$$

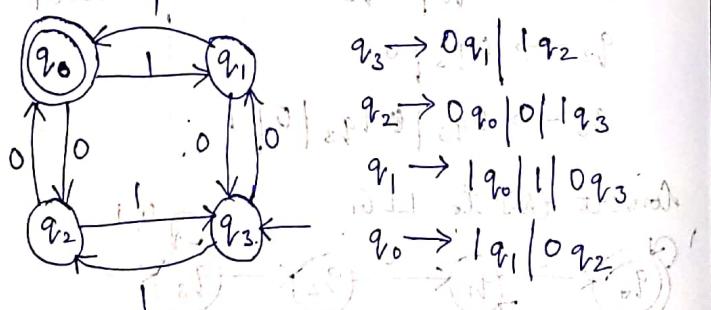
Now interchange,

$$\begin{aligned}
 q_3 &\rightarrow q_30 \mid q_31 \mid q_20 \\
 q_2 &\rightarrow q_10 \quad \text{as } q_2 \rightarrow (q_1, 0) \\
 q_1 &\rightarrow q_00 \mid 0 \\
 q_0 &\rightarrow q_01 \mid 1 \mid q_1
 \end{aligned}$$

Q1. Find LLG for the given DFA.



To convert into LLG, reverse the DFA.



$$\begin{aligned}
 \text{LLG} \Rightarrow \quad q_3 &\rightarrow q_10 \mid q_21 \\
 q_2 &\rightarrow q_00 \mid 0 \mid q_31 \\
 q_1 &\rightarrow q_01 \mid 1 \mid q_30 \\
 q_0 &\rightarrow q_11 \mid q_20
 \end{aligned}$$

Conversion of R.G to Finite Automata :-

1. First convert given LLG into RLG.
2. Take all possible suffixes of R.H.S of rule as states of FA.  
ex: In  $S \rightarrow 0IS$ , possible suffixes are  $[0IS], [IS], [S], [\epsilon]$ .
3. Draw FA with start state as initial state and  $[\epsilon]$  as final state.
4. If A is a non-terminal in the grammar and  $A \rightarrow \alpha$  is a production, then add the transition:

$$A \xrightarrow{\epsilon} \alpha$$

5. If  $a$  is a terminal and  $\alpha$  is some possible suffix of the form  $T^*$  or  $T^*V$ , then add a transition by  $S([ax], a) = [\alpha]$ .

$$ax \xrightarrow{a} \alpha$$

6. This NFA corresponds to RLG. If given NFA is LLG then follow 7 & 8 steps, otherwise stop.
7. Reverse the NFA  $M'$  to get  $M$  such that,
  - Initial state of  $M'$  is made final state of  $M$ .

- Final state of  $M'$  is initial state of  $M$ .
- Reverse all edges of  $M'$  and add to  $M$ .
- Obtained NFA  $M$  corresponds to given LLG.

$$① \quad S \rightarrow aA \mid b$$

$$A \rightarrow aA \mid a$$

Soln: This is R.L.G.  
Suffixes are,  $[aA]$ ,  $[A]$ ,  $[b]$ ,  $[a]$ .  
Rule 1 (R1):  
 $\delta([S], e) = [aA], [b]$ .

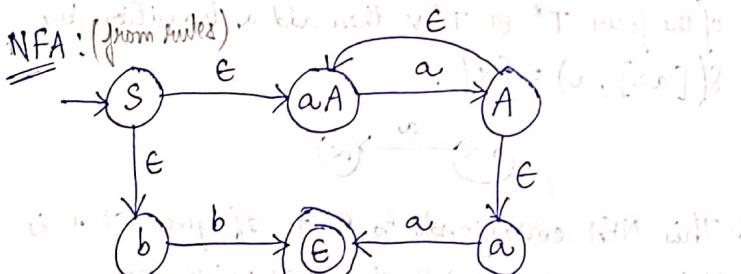
$$\delta([A], e) = [aA], [a]$$

Rule 2 (R2):

$$\delta([aA], a) = [A] \quad \text{from productions}$$

$$\delta([a], a) = [e] \quad \text{given in qstn}$$

$$\delta([b], b) = [e]$$



$$② \quad S \rightarrow Ab \mid ab$$

$$A \rightarrow Ab \mid Bb$$

$$B \rightarrow Ba \mid a$$

Soln: Since this is LLG, convert to R.L.G.  
 $S \rightarrow bA \mid ba$   
 $A \rightarrow bA \mid BB$   
 $B \rightarrow ab \mid a$   
Suffixes are,  $[bA]$ ,  $[A]$ ,  $[ba]$ ,  $[a]$ ,  
 $[bB]$ ,  $[B]$ ,  $[ab]$ .

$$\underline{R1:} \quad \delta([S], e) = [bA], [ba]$$

$$\delta([A], e) = [bA], [bb]$$

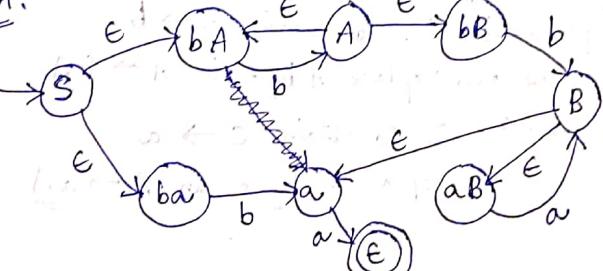
$$\delta([B], e) = [ab], [a]$$

$$\underline{R2:} \quad \delta([bA], b) = [A] \quad \delta([ba], b) = [a]$$

$$\delta([a], a) = e, \quad \delta([bB], b) = [B]$$

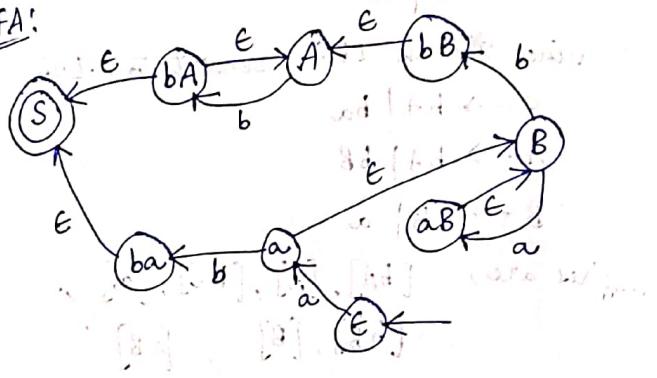
$$\delta([ab], a) = [B]$$

NFA:

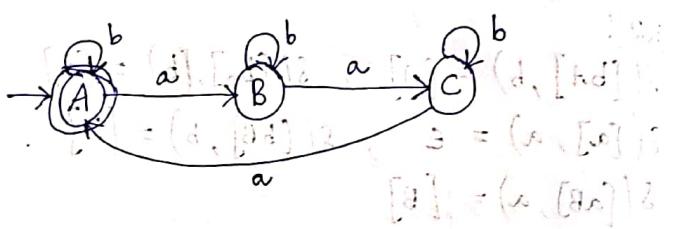


The obtained NFA is for R-LG. Now, it is LLG. But in Q3, it is LLG. So reverse the NFA.

$\therefore$  NFA:



① Draw the FA for the foll:  $A \rightarrow aB \mid bA \mid b$ ,  $B \rightarrow aC \mid bB$ ,  $C \rightarrow aA \mid bC \mid a$



A is final state because,

in production 1, to obtain  $A \rightarrow b$ , we can replace A by  $\epsilon$  in  $A \rightarrow bA$ .

in production 3, to obtain  $C \rightarrow a$ ,

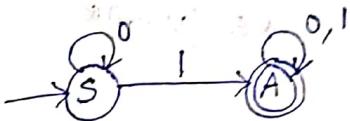
we can replace A by  $\epsilon$  in  $A \rightarrow aA$ .

If it is LLG.

LLG  $\rightarrow$  RG  $\rightarrow$  FA (Reverse)  $\rightarrow$  F

② Draw the FA for the foll:

$S \rightarrow OS \mid IA \mid I$ ,  $A \rightarrow OA \mid IA \mid O \mid I$

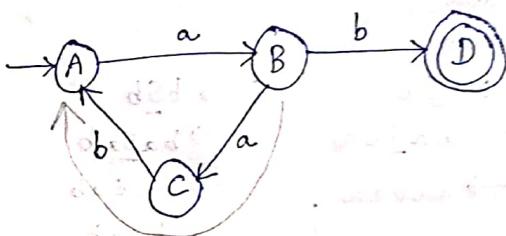


A is final state, to obtain  $S \rightarrow I$ , take  $S \xrightarrow{IA} I$

to obtain  $A \rightarrow O$ , take  $A \xrightarrow{OA} O$

to obtain  $A \rightarrow I$ , take  $A \xrightarrow{IA} I$

③  $A \rightarrow aB$ ,  $B \rightarrow abA \mid b$



C is taken for  $B \rightarrow abA$ .

D is taken for because,  $A \rightarrow ab \rightarrow ab$  [ $\because B \rightarrow b$ ].

Hence ab is the min. string to be accepted.

### Conversion of LLG to RLG:-

LLG  $\rightarrow$  R  $\xrightarrow{\text{Reverse}}$  FA  $\rightarrow$  F  $\rightarrow$  g (obtain grammar)

### Context-free Grammar:-

A grammar is context free if for  $(V, T, P, S)$  tuples

$$A \rightarrow x$$

where  $A \in V$ ,  $x \in (V \cup T)^*$

- ①:  $S \rightarrow aS$   
 $S \rightarrow e$

Soln:  $S \rightarrow aS$        $S \rightarrow aS$   
 $\rightarrow ae$        $\rightarrow aaS$   
 $\rightarrow ab$        $\rightarrow aat$   
 $\rightarrow aa$        $\rightarrow aa$

Hence lang. generated is  $a^*$ .

- ②:  $S \rightarrow e/a/b$   
 $S \rightarrow aSa$   
 $S \rightarrow bSb$

Soln:  $S \rightarrow aSa$        $S \rightarrow bSb$   
 $\rightarrow aaaSaa$        $\rightarrow baSab$   
 $\rightarrow aabaa$        $\rightarrow baeab$   
 $\rightarrow aba$        $\rightarrow baab$

Hence the lang. generated by this grammar is word and word-reverse i.e. Palindrome.

$$③: S \rightarrow SaSbS \mid SbSaS \mid e$$

Soln:  $S \rightarrow SaSbS$        $S \rightarrow SbSaS$   
 $\rightarrow ea, eb, e$        $\rightarrow SbSaSbSbSaS$   
 $\rightarrow ab$        $\rightarrow abaabbab$   
 $S \rightarrow SbSaS$        $S \rightarrow SbSaS$   
 $\rightarrow Eb, Ea, E$        $\rightarrow SbSaSbSbSaS$   
 $\rightarrow ba$        $\rightarrow abbaabba$

Hence the lang. generated by this grammar is no. of a's = no. of b's.

$$④: S \rightarrow aS$$

$$S \rightarrow bS$$

$$S \rightarrow a$$

$$S \rightarrow b$$

Soln:  $S \rightarrow aS$        $S \rightarrow aS$        $S \rightarrow aS$   
 $\rightarrow aa$        $\rightarrow ab$        $\rightarrow aaS$   
 $S \rightarrow bS$        $S \rightarrow bS$        $S \rightarrow bS$   
 $\rightarrow bb$        $\rightarrow ba$        $\rightarrow bba$   
 $S \rightarrow bS$        $S \rightarrow bS$        $S \rightarrow bS$   
 $\rightarrow bb$        $\rightarrow ba$        $\rightarrow bbb$

Hence the lang is  $(ab)^+$ .

- ① Generate grammar (context-free) for the R.E  $(011+1)^*(01)^*$

Soln: First consider  $(011+1)^*$   $\xrightarrow{c}$

$$A \rightarrow CA | E$$

$$C \rightarrow 011 | 1$$

Now consider  $(01)^*$

$$B \rightarrow DB | E$$

$$D \rightarrow 01$$

∴ Finally,  $S \rightarrow AB$

$$A \rightarrow CA | E$$

$$C \rightarrow 011 | 1$$

$$B \rightarrow DB | E$$

$$D \rightarrow 01$$

Check by constructing strings,

$$S \rightarrow \underline{AB}$$

$$\rightarrow \underline{CA}B$$

$$011EE$$

$$\rightarrow 011$$

② Construct the context-free grammar for which the 1st & last symbol differ where the inputs are {0,1}.

soln: The R-E is,

$$\Rightarrow 0(\underline{0+1})^*1 | 1(\underline{0+1})^*0$$

$$S \rightarrow 0A1 | 1A0$$

$A \rightarrow 0A | 1A | E$  [We can't write  $0|1$ , because we have  $\#$ ]

$$ex: S \rightarrow 0A1$$

$$\rightarrow 0E1$$

$$\rightarrow 01$$

③ Write the context-free grammar for

$$a^n b^{2n} | n \geq 1$$

soln:  $a^n b^{2n} \approx a^n (bb)^n$

$$S \rightarrow aSbb | abb$$

[∴ abb, aabb, aaabbbb, ... are acceptable.]

$$(4) a^m b^n | m, n \geq 1$$

$$soln: S \rightarrow aSb | a | b | ab | S$$

$$S \rightarrow aA | bB$$

$$A \rightarrow aA | a$$

$$B \rightarrow bB | b$$

### Derivations of Context-free Grammar:-

It is a process of defining the string out of grammar by application of rules starting from the start symbol.

ex: ①  $S \rightarrow aS$  } Derive  $a^4$  from given grammar  
 $= S \rightarrow E$

$$soln: S \rightarrow aS$$

$$\rightarrow aas$$

$$\rightarrow aaaS$$

$$\rightarrow aaaas$$

$$\rightarrow aaaaE$$

we will get  $a^4$ .

$$② \boxed{S \rightarrow aS | bs | a | b}$$

$$S \rightarrow aS$$

$$S \rightarrow bs$$

$$S \rightarrow a$$

$$S \rightarrow b$$

$$soln: S \rightarrow aS$$

$\rightarrow \underline{abS}$   
 $\rightarrow \underline{abbS}$   
 $\rightarrow \underline{abbaS}$   
 $\rightarrow abbab$

### Left most derivation:-

The left most non-terminal in a string is the first non-terminal that we encounter when we scan from left to right.

Replacing the left most variable first at every step gives the left most derivation.

ex:  $S \rightarrow baxas | ab$

$X \rightarrow Xab | aa$

Derive  $baaaabaab$  using left most derivation.

sol<sup>n</sup>:  $S \rightarrow baxas$

$\Rightarrow ba\underline{x}as$

$\Rightarrow ba\underline{aa}abas$

$\Rightarrow ba\underline{aa}abaab$

### Right most derivation:-

Replacing the right most variables first at every step gives the RMD.

ex:  $S \rightarrow baxas | ab$

$X \rightarrow Xab | aa$

Derive  $baaaabaab$  using RMD.

sol<sup>n</sup>:  $S \rightarrow baxas$

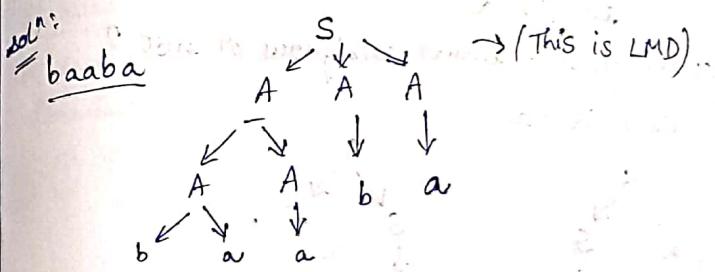
$\Rightarrow ba\underline{x}ab$

$\Rightarrow ba\underline{x}abab$   
 $\Rightarrow ba\underline{aa}abaab$

### Derivation Tree :-

Process shown pictorially as a tree to illustrate how a word derived from CFG. It is also called syntax trees / parse trees.

ex:  $S \rightarrow AAA | AA$  } Derive baaba  
 $A \rightarrow AA | aA | Ab | a | b$  }



### Ambiguous Grammar:-

If there exists more than one path stream or there exists more than one LMD or more than one RMD then the grammar is said to be ambiguous.

ex: ①  $E \rightarrow id | E+E | E^*E | E-E$

LMD : (lets derive  $id + id^* id$ )

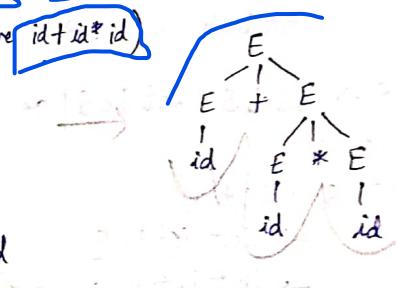
$E \Rightarrow E+E$

$\Rightarrow id + E$

$\Rightarrow id + E^*E$

$\Rightarrow id + id^*E$

$\Rightarrow id + id^*id$



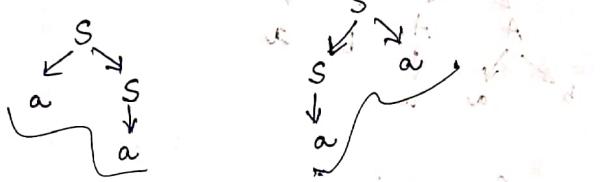
Now consider  $E \Rightarrow E^* E$

$$\begin{aligned} &\Rightarrow E + E^* E \\ &\Rightarrow id + E^* E \\ &\Rightarrow id + id^* E \\ &\Rightarrow id + id^* id \end{aligned}$$

For 2 strings we got same productions, so the given grammar is ambiguous. (Can prove using derivations or derivation tree).

②  $S \rightarrow aS | Sa | a$   
Is the given grammar ambiguous or not?

Soln: Let's derive aa.



Hence grammar is ambiguous.

③  $S \rightarrow aSa | bSb | aib | bi | cib | cbi$  Ambiguous or not?

Soln: Consider bab  
↳ can be derived in only 1 way  
i.e.  $S \rightarrow bSb$   
 $\rightarrow b \underline{ab}$  ✓ So, it is not ambiguous.

④  $S \rightarrow ictS | ictSeS | a$

$C \rightarrow b$   
Solv:  $S \rightarrow ictS$   
 $\rightarrow ict \underline{ictSeS}$   
 $\rightarrow ibt \underline{ibtaea}$  ✓

Now,  $S \rightarrow ictSeS$

$\rightarrow ict \underline{ictSeS}$

$\rightarrow ibt \underline{ibtaea}$  ✓

2 strings → same productions. So the grammar is ambiguous.  
Simplification / minimisation of grammar:-

3 steps :-

\* Elimination of useless symbols.

\* Elimination of E production

\* Elimination of unit productions like  $A \rightarrow B$  both A, B non-terminals.

Elimination of useless symbols :-

A symbol is useless if it cannot derive a terminal or is not reachable from start symbol.

ex: ①  $\begin{aligned} S &\rightarrow ABA | BC \\ A &\rightarrow aC | BCC \\ C &\rightarrow a \\ B &\rightarrow bcc \\ D &\rightarrow E \\ E &\rightarrow d \\ F &\rightarrow e \end{aligned}$

Eliminate useless symbols.

Soln:  $S - A - B - C$

$\underline{D} \quad \underline{E} \quad \underline{F}$

So finally  $\begin{aligned} S &\rightarrow ABA | BC \\ A &\rightarrow ac | BCC \\ C &\rightarrow a \\ B &\rightarrow bcc \end{aligned}$  after removing useless symbols.

⑤  $\begin{aligned} S &\rightarrow AB | CA \\ S &\rightarrow BC | AB \\ A &\rightarrow a \end{aligned}$

$$C \rightarrow ab \mid b$$

Soln: There is no production from B. So it is useless.  
So after removing productions containing B,

$$S \rightarrow CA$$

$$A \rightarrow a$$

$$C \rightarrow b$$

$$(3) S \rightarrow aS \mid A \mid BC$$

$$A \rightarrow a$$

$$B \rightarrow aa$$

$$C \rightarrow acb$$

Soln: Extra empty string can't be added.

$$\text{Finally, } S \rightarrow aS \mid A$$

$$A \rightarrow a$$

Elimination of  $\epsilon$  (E) production:-

$$(1) S \rightarrow ABaC$$

$$A \rightarrow BC$$

$$B \rightarrow b \mid \epsilon$$

$$C \rightarrow D \mid \epsilon$$

$$D \rightarrow d$$

1. Nullable variables. {B, C, A}.

$$S \rightarrow ABaC \mid BaC \mid AaC \mid ABa \mid ac \mid Aa \mid Ba \mid a$$

Changing to non- $\epsilon$  putting A, B, C as  $\epsilon$ .

$$A \rightarrow BC \mid C \mid B$$

$$B \rightarrow b$$

$$C \rightarrow D$$

$$D \rightarrow d$$

$$(2) S \rightarrow aA$$

$$A \rightarrow BB$$

$$B \rightarrow aBb \mid \epsilon$$

Soln: Nullable variables. {B, A}.

$$S \rightarrow aA \mid a$$

$$A \rightarrow BB \mid B$$

$$B \rightarrow aBb \mid ab$$

Elimination of unit production:-

$$(1) S \rightarrow A \mid bb$$

$$A \rightarrow B \mid b$$

$$B \rightarrow S \mid a$$

$$\text{Soln: } S \rightarrow a \mid b \mid bb$$

$$\hookrightarrow (S \rightarrow A \rightarrow B \rightarrow a)$$

$$A \rightarrow a \mid b \mid bb$$

$$\hookrightarrow (A \rightarrow B \rightarrow a \rightarrow b)$$

$$B \rightarrow a \mid b \mid bb$$

$$(2) S \rightarrow Aa \mid B$$

$$B \rightarrow A \mid bb$$

$$A \rightarrow a \mid bc \mid B$$

$$\text{Soln: } S \rightarrow Aa \mid bb \mid a \mid bc$$

$$B \rightarrow bb \mid a \mid bc$$

$$A \rightarrow a \mid bc \mid bb$$

$$(3) S \rightarrow aA \mid aBB$$

$$A \rightarrow aAA \mid \epsilon$$

$$B \rightarrow bb \mid bbb$$

$$C \rightarrow B$$

Soln: Eliminating null production,

$$S \rightarrow aA \mid a \mid aBB$$

$$A \rightarrow aAA \mid aA \mid a$$

$$B \rightarrow bb \mid bbb$$

$$C \rightarrow B$$

3 soln: (cont).

eliminate useless symbols  
(B, C because they don't derive anything).

$$S \rightarrow aA | a$$

$$A \rightarrow aAA | aA | a$$

→ No unit productions involved; so this is the final grammar.

Elimination of Left Recursion:-

If production is of form  $A \rightarrow A\alpha | B\beta$ ,

then add  $A \rightarrow BA'$

$$\begin{array}{|l} A \rightarrow BA' \\ A' \rightarrow \alpha A' | \epsilon \end{array}$$

ex: ①  $\begin{array}{|l} E \rightarrow E + T | T \\ A \quad A \alpha \quad B \end{array}$

$$A \rightarrow E$$

$$\alpha \rightarrow +T$$

$$B \rightarrow T$$

$$\therefore S \rightarrow E \rightarrow TE' \rightarrow E + TE' | \epsilon$$

②  $S \rightarrow S \alpha S | \alpha S | \alpha$

$$\alpha = \alpha S S$$

$$B = \alpha$$

$$\therefore S \rightarrow \alpha S | \alpha$$

$$S' \rightarrow \alpha S S' | \epsilon$$

③  $S \rightarrow (L) | \alpha$

$$L \rightarrow L, S | S$$

$$\therefore \begin{array}{c} L \rightarrow SL' \\ L' \rightarrow , SL' | \epsilon \end{array} \rightarrow S \rightarrow (L) | \alpha \quad \begin{array}{l} \text{(No change} \\ \text{bcz} \\ \text{no LR)} \end{array}$$

If production is of the form  $A \rightarrow A\alpha | B_1\beta_1 | B_2\beta_2 \dots$

then add

$$\begin{array}{|l} A \rightarrow B_1 A' | B_2 A' \\ A' \rightarrow \alpha A' | \epsilon \end{array}$$

If production is of the form  $A \rightarrow A\alpha_1 | A\alpha_2 | \dots | A\alpha_n | B$ ,

then add

$$\begin{array}{|l} A \rightarrow B A' \\ A' \rightarrow \alpha_1 A' | \alpha_2 A' | \dots | \alpha_n A' | \epsilon \end{array}$$

Elimination of Left Factoring:-

$$S \rightarrow aA | aB$$

NDG  $\rightarrow$  DG

(Non-deterministic grammar  $\rightarrow$  deterministic graph)

ex: ①  $S \rightarrow aS | aB | a | b$

If  $A \rightarrow \alpha B_1 | \alpha B_2 | \alpha B_3 | \gamma$

add  $\begin{array}{|l} A \rightarrow \alpha A' | \gamma \\ A' \rightarrow B_1 | B_2 | B_3 \end{array}$

$$\Rightarrow S \rightarrow aS' | a | b$$

$$S' \rightarrow S | B$$

$$② A \rightarrow aSb \mid aSc \mid b$$

Soln:

$$\begin{aligned} A &\rightarrow aSA' \mid b \\ A' &\rightarrow S \mid b/c \end{aligned}$$

Normal forms:-

i. Chomsky Normal Form (CNF) :-

A CFG is in CNF if each of the production has one of the 2 forms.

① Should contain 2 variables on right side.  
ex:  $A \rightarrow BC$ .

② Should contain 1 terminal  
ex:  $A \rightarrow aab$ .

Ex: ① Convert the foll: grammar to CNF.

$$A \rightarrow bAA \mid aS \mid a$$

$$B \rightarrow abb \mid bS \mid b$$

Soln: (i) replacing every terminal by variable.

~~Converting to CNF~~

$$\begin{aligned} S &\rightarrow C_b A' \mid C_a B \\ A &\rightarrow C_b AA \mid C_a S \mid a \end{aligned}$$

$$B \rightarrow C_a BB \mid C_b S \mid b$$

$$C_a \rightarrow a$$

$$C_b \rightarrow b$$

Now converting those which are not in CNF.

$$S \rightarrow C_b A \mid C_a B$$

(No change bcz it is in CNF)

$$A \rightarrow C_b D \mid C_a S \mid a$$

$$B \rightarrow C_a E \mid C_b S \mid b$$

$$C_a \rightarrow a$$

$$C_b \rightarrow b$$

$$D \rightarrow AA$$

$$E \rightarrow BB$$

$$S \rightarrow AB \mid aB$$

$$A \rightarrow aab \mid e$$

$$B \rightarrow bbA$$

Soln: Removing E production,

$$S \rightarrow AB \mid B \mid ab$$

$$A \rightarrow aab$$

$$B \rightarrow bbA \mid bb$$

Removing unit productions,

$$S \rightarrow AB \mid bbA \mid bb \mid ab$$

$$A \rightarrow aab$$

$$B \rightarrow bbA \mid bb$$

$$S \rightarrow AB \mid C_b C_b A \mid C_b C_b \mid C_a B$$

$$A \rightarrow C_a C_a B$$

$$B \rightarrow C_b C_b A \mid C_b C_b$$

$$C_a \rightarrow a$$

$$C_b \rightarrow b$$

$$S \rightarrow AB \mid C_{bb} A \mid C_b C_b \mid C_a B$$

$$A \rightarrow C_{aa} C_b$$

$$B \rightarrow C_{bb}A \mid C_bC_b$$

$$C_a \rightarrow a$$

$$C_b \rightarrow b$$

$$C_{aa} \rightarrow C_aC_a$$

$$C_{bb} \rightarrow C_bC_b$$

$$S' \rightarrow S$$

$$S \rightarrow ASB \mid E$$

$$A \rightarrow aAS \mid a$$

$$B \rightarrow SbS \mid A \mid bb$$

Soln: Removing E productions,

$$S \rightarrow ASB \mid AB$$

$$A \rightarrow aAS \mid aA \mid a$$

$$B \rightarrow SbS \mid bS \mid Sb \mid b \mid A \mid bb$$

Removing unit productions,

$$S \rightarrow ASB \mid AB$$

$$A \rightarrow aAS \mid aA \mid a$$

$$B \rightarrow SbS \mid bS \mid Sb \mid b \mid aAS \mid aA \mid a \mid bb$$

$$S \rightarrow DB \mid AB$$

$$A \rightarrow C_aD \mid C_aA \mid a$$

$$B \rightarrow SC_bS \mid C_bS \mid SC_b \mid b \mid C_aD \mid C_aA \mid a \mid C_bC_b$$

$$C_a \rightarrow a$$

$$C_b \rightarrow b$$

$$D \rightarrow AS$$

$$S \rightarrow DB \mid AB$$

$$A \rightarrow C_aD \mid C_aA \mid a$$

$$B \rightarrow ES \mid C_bS \mid SC_b \mid b \mid C_aD \mid C_aA \mid a \mid C_bC_b$$

$$C_a \rightarrow a$$

$$C_b \rightarrow b$$

$$D \rightarrow AS$$

$$E \rightarrow SC_b$$

Procedure to convert CFG to CNF :-

1. Eliminating null and unit productions.
2. Include the productions of the form  $A \rightarrow BC \mid a \dots$  as it is.
3. Eliminate string of terminals on RHS of production if it exists/agrees one of the following:

$$S \rightarrow a_1a_2a_3 \quad (c_1 \rightarrow a_1, c_2 \rightarrow a_2, c_3 \rightarrow a_3)$$

Introduce non-terminals for every terminal.

4. To restrict the no. of variables on the RHS, if the production is  $Y \rightarrow X_1X_2X_3X_4 \dots$

$$Y \rightarrow X_1R \quad (R = X_2X_3X_4 \dots)$$

$$R \rightarrow X_2R_1 \quad (R_1 = X_3X_4 \dots)$$

$$R_1 \rightarrow X_3X_4$$

We need to include new variables by combining two-two variables.

Can write in anyway but RHS should contain 2 variables

Greibach normal form :- (GNF) :-  
A CFG is said to be in GNF if all the productions are in the form of

$$A \rightarrow aX$$

where  $a \in T$ ,  $X \in V^*$  (any no. of variables).

ex:  $A \rightarrow a$   $\quad A \rightarrow ab$

Procedure to convert CFG to GNF:

1. Eliminate null and unit productions and construct CNF.

2. Rename the variables (like  $A_1, A_2, A_3, \dots$ ).

3. For production  $A_i \rightarrow A_j X$  or  $(G \rightarrow XB) \quad (S = A_1, B = A_2, X = A_3)$

- (i)  $j > i \rightarrow$  leave as it is.
- (ii)  $j = i \rightarrow$  apply left recursion rule.
- (iii)  $j < i \rightarrow$  substitution rule

Q1.

$$S \rightarrow XA | BB$$

$$B \rightarrow b | SB$$

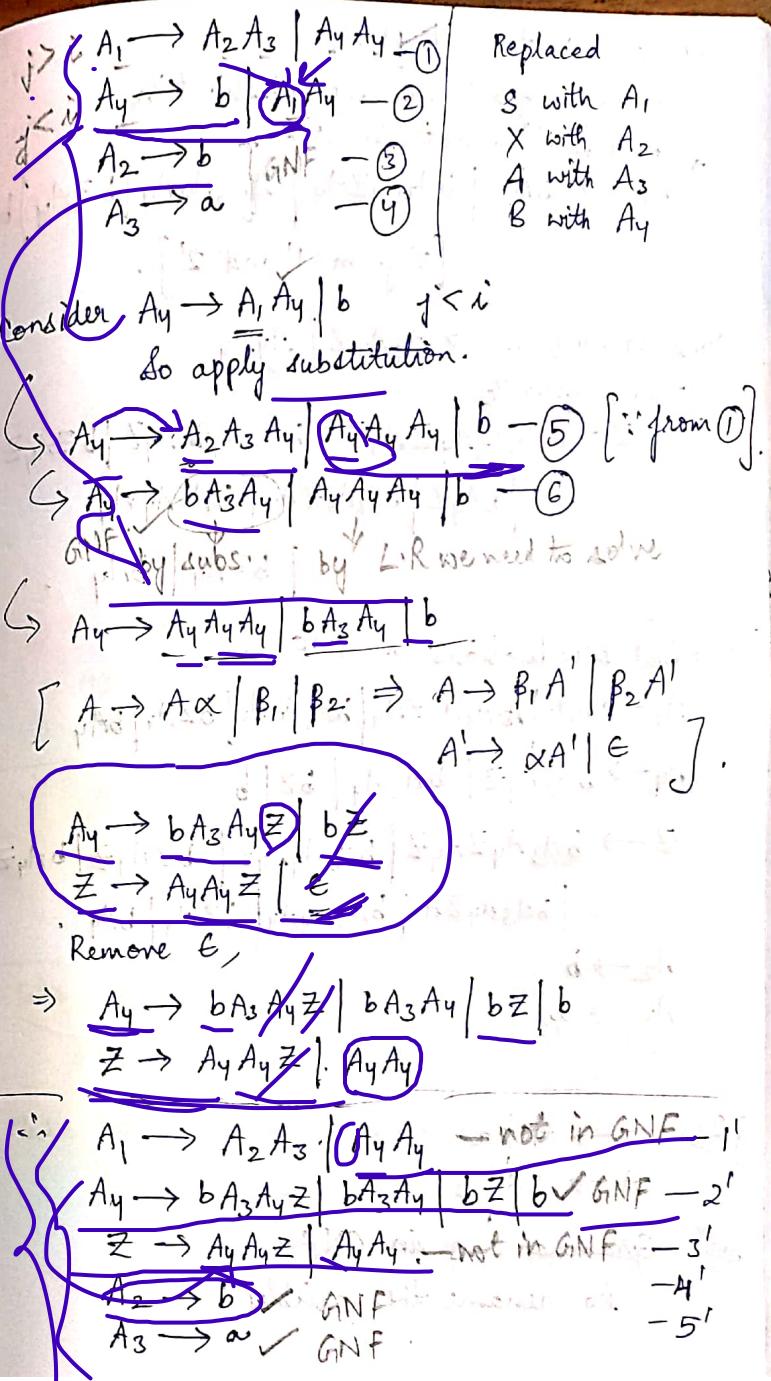
$$X \rightarrow b | XA$$

$$A \rightarrow a$$

Soln: No null and unit productions.

The grammar is already in CNF.

Now, Rename the variables,





For  $A_1 \rightarrow \underline{A_2 A_3} \mid \underline{A_4 A_4}$   
even though  $j > i$  we use substitution  
because it is not in GNF.

$$A_1 \rightarrow \underline{bA_3} \mid bA_3A_4ZA_4 \mid bA_3A_4A_4 \mid bZA_4 \mid bA_4 \\ [\because \text{from 1' and 2'}]$$

Consider now,  $Z \rightarrow \underline{A_4 A_4 Z} \mid \underline{A_4 A_4}$ .

Now substitute,  
 $Z \rightarrow bA_3A_4ZA_4Z \mid bA_3A_4ZA_4Z \mid bZA_4Z \\ \mid bA_4Z \mid bA_3A_4ZA_4 \mid bA_3A_4A_4 \\ \mid bZA_4 \mid bA_4$

∴ Final productions in GNF are - - -

$$A_1 \rightarrow bA_3 \mid bA_3A_4ZA_4 \mid bA_3A_4A_4 \mid bZA_4 \mid bA_4$$

$$A_4 \rightarrow bA_3A_4Z \mid bA_3A_4 \mid bZ \mid b$$

$$Z \rightarrow bA_3A_4ZA_4Z \mid bA_3A_4A_4Z \mid bZA_4Z \mid bA_4Z \\ \mid bA_3A_4ZA_4 \mid bA_3A_4A_4 \mid bZA_4 \mid bA_4$$

$$A_2 \rightarrow b$$

$$A_3 \rightarrow a$$

Q2:  $S \rightarrow AA \mid a$

$$A \rightarrow SS \mid b$$

Soln: Grammar is in CNF.  
Do rename the variables.

$$\begin{array}{l} j > i: A_1 \rightarrow A_2 A_2 \mid a \quad \text{--- 1} \\ j < i: A_2 \rightarrow A_1 A_1 \mid b \quad \text{--- 2} \end{array} \quad \begin{array}{l} \text{Replaced} \\ S \text{ with } A_1 \\ A \text{ with } A_2 \end{array}$$

Consider  $A_2 \rightarrow A_1 A_1 \mid b$   $j < i$   
So apply substitution.

$$A_2 \rightarrow \underline{A_2 A_2} \mid \underline{A_1 A_1} \mid b$$

$$\left[ \begin{array}{l} A \rightarrow A\alpha \mid \beta_1 \mid \beta_2 \Rightarrow A \rightarrow \beta_1 A^1 \mid \beta_2 A^1 \\ A^1 \rightarrow \alpha A^1 \mid \epsilon \end{array} \right]$$

$$A_2 \rightarrow aA_1Z \mid bZ$$

$$Z \rightarrow A_2 A_1 Z \mid \epsilon$$

Remove  $\epsilon$ ,

$$A_2 \rightarrow aA_1Z \mid aA_1 \mid bZ \mid b$$

$$Z \rightarrow A_2 A_1 Z \mid A_2 A_1$$

$\therefore A_1 \rightarrow A_2 A_2 \mid a$  ~~not in GNF~~ --- 1

$$A_2 \rightarrow aA_1Z \mid aA_1 \mid bZ \mid b$$
 ~~CNF~~ --- 2

$$Z \rightarrow A_2 A_1 Z \mid A_2 A_1$$
 ~~not in GNF~~ --- 3

Take  $A_1 \rightarrow \underline{A_2 A_2} \mid a$  Substitution

$$A_1 \rightarrow aA_1Z A_2 \mid aA_1 A_2 \mid bZA_2 \mid bA_2 \mid a$$

Take  $Z \rightarrow \underline{A_2 A_1 Z} \mid \underline{A_2 A_1}$  Substitution

$$\begin{aligned} Z \rightarrow aA_1Z A_1 Z \mid aA_1 A_1 Z \mid bZA_1 Z \mid bA_1 Z \\ \mid aA_1 Z A_1 \mid aA_1 A_1 \mid bZA_1 \mid bA_1 \end{aligned}$$

So  $aA_1 Z A_1 \mid aA_1 A_1 \mid bZA_1 \mid bA_1$

So  $aA_1 Z A_1 \mid aA_1 A_1 \mid bZA_1 \mid bA_1$

$\therefore$  Final productions in GNF are,

$$A_1 \rightarrow aA_1ZA_2 \mid aA_1A_2 \mid bZA_2 \mid bA_2 \mid a$$

$$A_2 \rightarrow aA_1Z \mid aA_1 \mid bZ \mid b$$

$$Z \rightarrow aA_1ZA_1Z \mid aA_1A_1Z \mid bZA_1Z \mid bA_1Z$$

$$\quad \quad \quad bA_1ZA_1 \mid aA_1A_1 \mid bZA_1 \mid bA_1$$

Pumping Lemma for CFG :-

If  $A$  is a CFG then  $A$  has pumping length  $p$  such that the string is divided into 5 features/pieces.

$$S = uvxyz$$

such that,

- (i)  $uv^ixyz \in A$  for all  $i \geq 0$
- (ii)  $|vxy| > 0$
- (iii)  $|vxy| \leq p$

$$\text{Q1. } S = a^n b^n c^n \mid n \geq 0$$

Prove that given lang. is not context-free.

Soln: We prove by contradiction i.e. assuming that it is CFG if true then  $\therefore$  it is CFG.

Take  $n=4$ ,  $\underline{\text{aaaa}}. \underline{\text{b.bbb}}. \underline{\text{cccc}}$  (can take in any manner).

Take  $i=3$ ,  $uv^iz = \underline{\text{aaaabb}} \underline{\text{bbb}} \underline{\text{bbb}} \underline{\text{cccccc}}$   $\notin A$ .

P.L 1st condition failed.

$\therefore$  The given grammar is not CFG.

[Also we took  $p=3$  such that (ii)(iii) are satisfied bcz  $|vxy| = 2 > 0$   
 $|vxy| = 3 = p = 3 \checkmark$  ].

Closure Properties of CFL (Context-free languages) :-

- ① CFL are closed under union.
  - ② CFL are closed under concatenation.
  - ③ CFL are closed under closure.
  - ④ They are not closed under intersection.
  - ⑤ They are not closed under compliment.
- [Explanation same as given for properties of CFG]

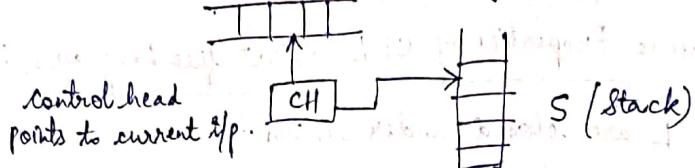
→ State transition diagram

12/7/21

## Unit - 4

Push Down Automata (PDA):-

Since there are a few dis. adv. of FA we go for PDA.  
IT (Input Type)



There are 7 tuples.

$$M = (Q, \Sigma, \delta, \gamma, q_0, z_0, F)$$

FA dis. adv.  $\rightarrow$  very finite amount of memory.

$a^n b^n | n > 0 \rightarrow$  cannot construct finite automata.

PDA is a FA with extra memory called stack.

$$\Rightarrow M = (Q, \Sigma, \delta, \gamma, q_0, z_0, F)$$

Q - no. of states.

$\Sigma$  - input alphabet.

$\delta$  - transition function.

$\gamma$  - stack alphabet.

$q_0$  - initial state.

$z_0$  - initial stack symbol.

F - set of accepting states.

PDA moves:-

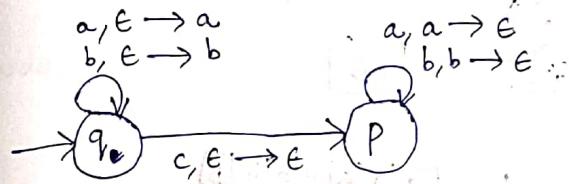
- Element to be added to the stack.
- Element may be deleted from the stack.

$$(q_0, a, z_0) = (q_0, \epsilon) \rightarrow \text{deleted}$$

(iii) They may (or) may not change their state.  
 $\delta(q_0, a, z_0) = (q_0, az_0)$   
 $\downarrow$  i/p symbol      initial stack symbol       $\boxed{\frac{a}{z_0}}$

Graphical Representation of PDA :-

$$\begin{aligned} \delta(q, a, \epsilon) &= (q, a) & Q - \{q, p\} \\ \delta(q, b, \epsilon) &= (q, b) & \Sigma - \{a, b, c\} \\ \delta(q, c, \epsilon) &= (p, \epsilon) & \gamma - \{a, b\} \\ \delta(p, a, a) &= (p, \epsilon) & q_0 - \{q\} \\ \delta(p, b, b) &= (p, \epsilon) & F - \{p\} \end{aligned}$$



Language acceptance by PDA :-

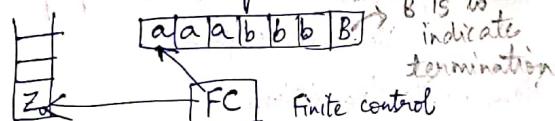
- ① When PDA reaches final state.      conditions  
 ② When PDA reaches empty stack.      for acceptance  
 (any 1 may be true).  
 (ie if stack is empty).

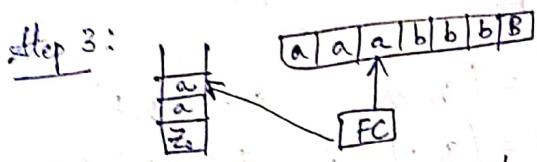
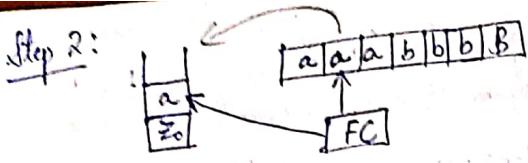
Ex: 1. Design a PDA which accepts  $a^n b^n | n \geq 1$ .

$$\text{soln: } L = \{a^n b^n | n \geq 1\}.$$

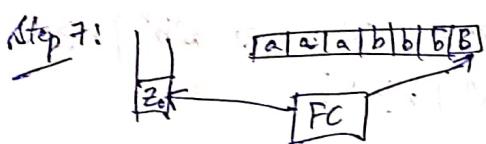
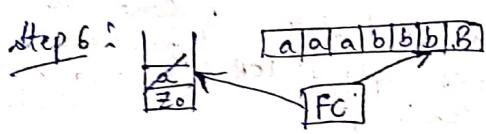
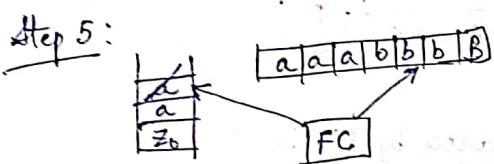
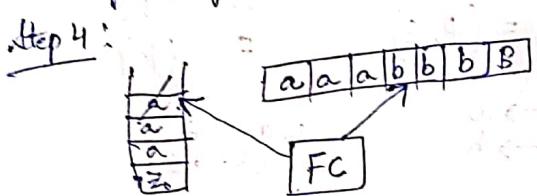
Let's consider the string : aaabbb

Step 1:





For every input 'a' we added 'a' into stack.  
Now for every input 'b' pop 'a' from stack.



Reached final state & stack is empty. So stop.  
Hence PDA accepted.

Now, write transitions :-

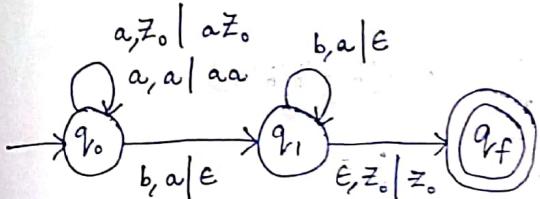
$$\delta(q_0, a, z_0) = (q_0, az_0)$$

i/p  $\delta(q_0, a, a) = (q_0, aa)$

Top of stack  $\delta(q_0, b, a) = (q_1, \epsilon)$

$$\delta(q_1, b, a) = (q_1, \epsilon)$$

$$\delta(q_1, \epsilon, z_0) = (q_f, z_0)$$



Consider aabb ✓.

abb X

ba X

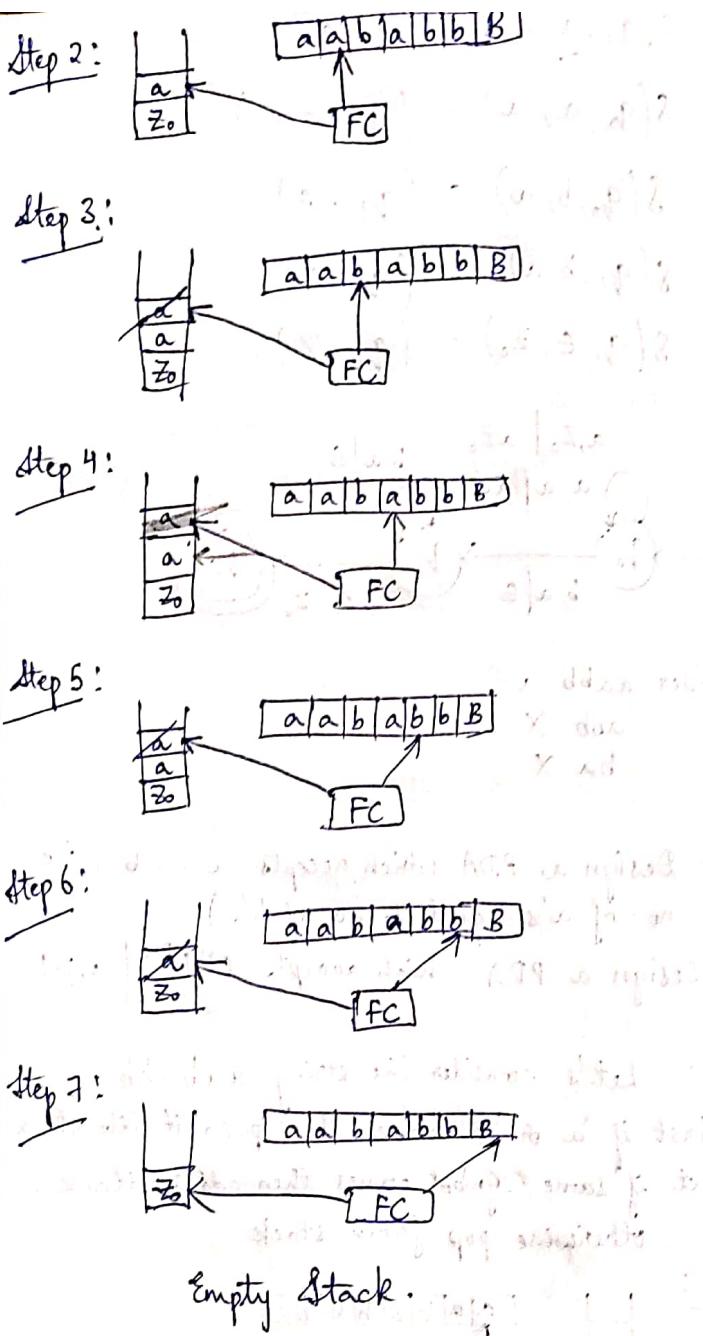
H.W. ① Design a PDA which accepts  $a = b$  (i.e. no. of a's equal to no. of b's):

② Design a PDA which accepts  $0^n 1^{2n} \mid n \geq 1$

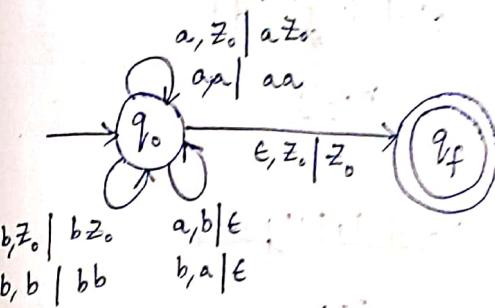
(1) sol<sup>n</sup>: Let's consider the string aabb.  
First if 'a' or 'b' comes then push it into stack.  
Next if same symbol comes then add to stack.  
otherwise pop from stack.

Step 1:





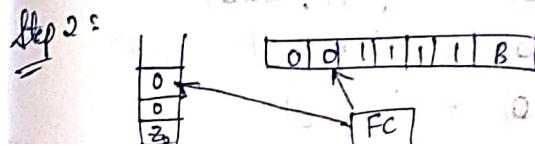
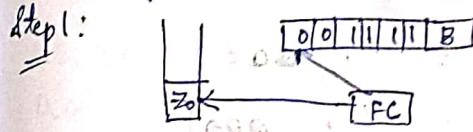
$$\begin{aligned}\delta(q_0, a, Z_0) &= (q_0, aZ_0) \\ \delta(q_0, a, a) &= (q_0, aa) \\ \delta(q_0, b, Z_0) &= (q_0, bZ_0) \\ \delta(q_0, b, b) &= (q_0, bb) \\ \delta(q_0, a, b) &= (q_0, \epsilon) \\ \delta(q_0, b, a) &= (q_0, \epsilon) \\ \delta(q_0, \epsilon, Z_0) &= (q_f, Z_0).\end{aligned}$$



(2) Let  $L = \{0^n 1^{2n} \mid n \geq 1\}$ . Consider the string 001111.

For every input '0', add '00' into stack.

For every input '1', pop '0' from stack.



② Sol<sup>n</sup>:  $L = \{0^n 1^{2n} \mid n \geq 1\}$

$$\delta(q_0, 0, z_0) = (q_0, 0z_0)$$

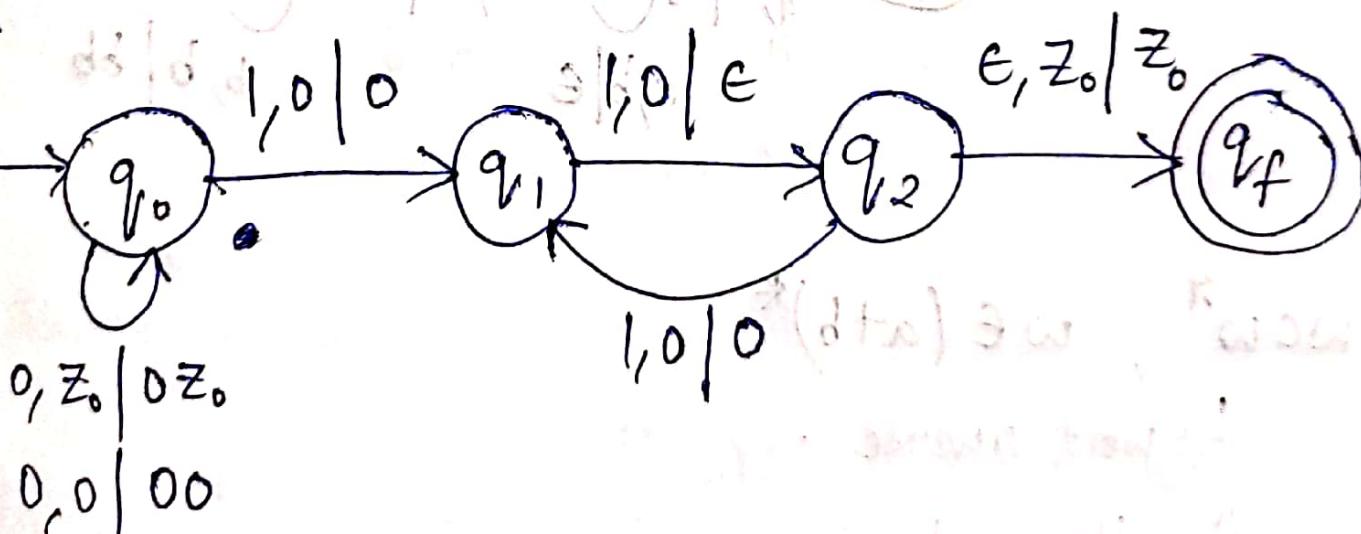
$$\delta(q_0, 0, 0) = (q_0, 00)$$

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

$$\delta(q_1, 1, 0) = (q_2, \epsilon)$$

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

$$\delta(q_2, \epsilon, z_0) = (q_f, z_0)$$



This is the PDA which accepts  $0^n 1^{2n} \mid n \geq 1$ .

Q. Design a PDA which accepts  
 $L = \{a^3 b^n c^n \mid n \geq 0\}$ .

$$\text{soln: } \delta(q_0, a, z_0) = (q_1, z_0)$$

$$\delta(q_1, a, z_0) = (q_2, z_0)$$

$$\delta(q_2, a, z_0) = (q_3, z_0)$$

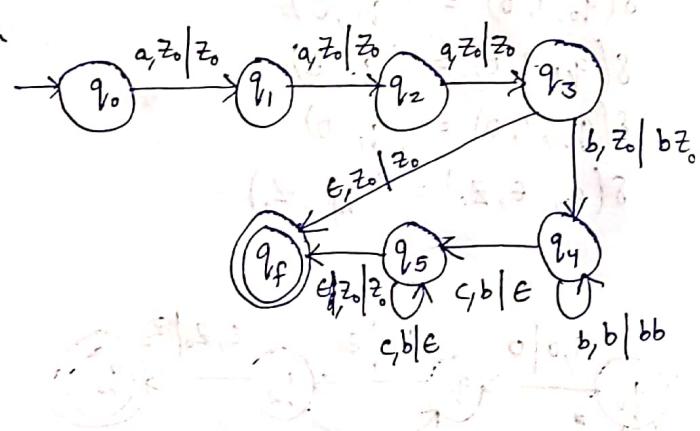
$$\delta(q_3, a, z_0) = (q_4, b z_0)$$

$$\delta(q_4, b, z_0) = (q_5, b b)$$

$$\delta(q_5, c, b) = (q_6, \epsilon)$$

$$\delta(q_6, c, b) = (q_7, \epsilon)$$

$$\delta(q_7, \epsilon, z_0) = (q_f, z_0)$$



Q.  $wcw^*$ ,  $w \in (a+b)^*$ .  
 word reverse.

soln: Consider abcba ✓

$$\delta(q_0, a, z_0) = (q_0, a z_0)$$

$$\delta(q_0, a, a) = (q_0, aa)$$

$$\delta(q_0, a, b) = (q_0, ab)$$

$$\delta(q_0, b, z_0) = (q_0, bz_0)$$

$$\delta(q_0, b, b) = (q_0, bb)$$

$$\delta(q_0, b, a) = (q_0, ba)$$

$$\delta(q_0, c, z_0) = (q_1, z_0)$$

$$\delta(q_0, c, a) = (q_1, a)$$

$$\delta(q_0, c, b) = (q_1, b)$$

$$\delta(q_1, a, a) = (q_1, \epsilon)$$

$$\delta(q_1, b, b) = (q_1, \epsilon)$$

$$\delta(q_1, \epsilon, z_0) = (q_f, z_0)$$

Equivalence of acceptance of final state and empty stack :-

$$L = a^n b^n$$

$$\delta(q_0, a, z_0) \rightarrow (q_0, a z_0)$$

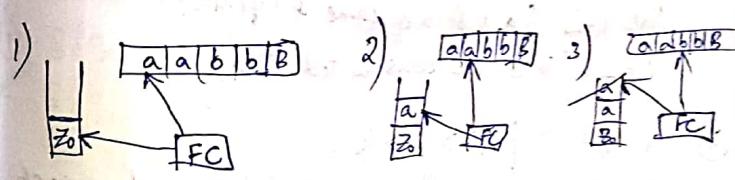
$$\delta(q_0, a, a) \rightarrow (q_0, aa)$$

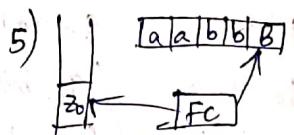
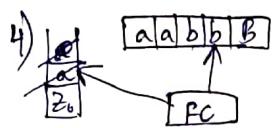
$$\delta(q_0, b, a) \rightarrow (q_1, \epsilon)$$

$$\delta(q_1, b, a) \rightarrow (q_1, \epsilon)$$

$$\delta(q_1, \epsilon, z_0) \rightarrow (q_f, z_0)$$

assume aabb.





Empty stack.

Final state :- ~ same as transition diagram  
(can write in any manner).

State (s)	<u>i/p</u>	<u>transition</u>	<u>top</u>	<u>States</u>
	aabb	(t)		(after move)
q0	a	①	$z_0$	$(q_0, q_0)$
q0	a	②	a	$(q_0, q_0)$
q0	b	③	a	$q_1$
q1	b	④	a	$q_1$
q1	$\epsilon$		$z_0$	$q_f$

Find whether given lang. are R.L., CFL, or not?

- ①  $a^k \mid k \text{ is even}$   
R.L. ✓ (we have finite automata).  
CFL ✓ bcz it is R.L.

- ②  $a^n b^n \mid n \geq 0$

Not R.L. (a's & b's same & infinite).

CFL ✓ (bcz we constructed PDA).

- ③  $a^i b^j c^k \mid i > j > k$

Not Regular L. because i, j, k depend on each other.

Not CFL (as there are 3 comparisons, we can't construct PDA).

④  $a^i b^j c^k \mid j = i+k$

Not R.L. (i, j, k depend on each other).

$$a^i b^{i+k} c^k \rightarrow \frac{a^i b^i}{b^k c^k}$$

push  $b^i$  for  $a^i$       push  $b^k$  for  $c^k$   
pop  $b^i$  for  $b^i$       pop  $b^k$  for  $c^k$

Hence it is CFL.

Constructing PDA for given CFG :-

①  $S \rightarrow aAA$

$A \rightarrow aS \mid bS \mid a$

$\Rightarrow S \rightarrow aAA \vee S(q, a, S) \rightarrow (q, AA)$

$A \rightarrow aS \mid bS \mid a \vee S(q, a, A) \rightarrow (q, S)$

$S(q, b, A) \rightarrow (q, S)$

$S(q, a, A) \rightarrow (q, \epsilon)$

Conversion of PDA to CFG :-

①  $\delta(q_0, b, z_0) = (q_0, zz_0) - 1$

$\delta(q_0, \epsilon, z_0) = (q_0, \epsilon) - 2$

$\delta(q_0, b, z) = (q_0, zz) - 3$

$\delta(q_0, a, z) = (q_0, z) - 4$

$$\delta(q_1, b, z) = (q_1, \epsilon) - 5$$

$$\delta(q_1, a, z_0) = (q_0, z_0) - 6$$

Defn: First we add S transition for every problem.

$$S \rightarrow [q_0, z_0, q_0] \mid [q_0, z_0, q_1]$$

[As there are  $q_0$  &  $q_1$  states overall and initially  $z_0$  on top of stack].

Now consider ①,

$$\delta(q_0, b, z_0) = (q_0, zz_0) \xrightarrow{2, z_0} \text{we get}$$

$$[q_0, z_0, q_0] \rightarrow b[q_0, zq_0] \quad [q_0, z_0, q_0]$$

$$[q_0, z_0, q_0] \rightarrow b[q_0, zq_1] \quad [q_1, z_0, q_0]$$

$$[q_0, z_0, q_1] \rightarrow b[q_0, zq_0] \quad [q_0, z_0, q_1]$$

$$[q_0, z_0, q_1] \rightarrow b[q_0, zq_1] \quad [q_1, z_0, q_1]$$

commas not a problem (better remove commas).

$$② \delta(q_0, \epsilon, z_0) = (q_0, \epsilon)$$

$$[q_0, z_0, q_0] \rightarrow \epsilon$$

$$③ \delta(q_0, b, z) = (q_0, zz)$$

$$[q_0, z, q_0] \rightarrow b[q_0, zq_0] \quad [q_0, zq_0]$$

$$[q_0, z, q_0] \rightarrow b[q_0, zq_1] \quad [q_1, zq_0]$$

$$[q_0, z, q_1] \rightarrow b[q_0, zq_0] \quad [q_0, zq_1]$$

$$[q_0, z, q_1] \rightarrow b[q_0, zq_1] \quad [q_1, zq_1]$$

$$④ \delta(q_0, a, z) = (q_1, z) \\ [q_0, zq_0] \rightarrow a[q_1, zq_0] \\ [q_0, zq_1] \rightarrow a[q_1, zq_1]$$

$$⑤ \delta(q_1, b, z) = (q_1, \epsilon) \\ [q_1, z, q_1] \rightarrow b$$

$$⑥ \delta(q_1, a, z_0) = (q_0, z_0) \\ [q_1, z_0, q_0] \rightarrow a[q_0, z_0, q_0] \\ [q_1, z_0, q_1] \rightarrow a[q_0, z_0, q_1]$$

Now rename the variables.

$q_0, z_0, q_0$  to A,  $q_0, z_0, q_1$  to B,  $q_1, z_0, q_0$  to C,

$q_1, z_0, q_1$  to D,  $q_0, zq_0$  to E,  $q_0, zq_1$  to F,

$q_1, zq_1$  to G,  $q_1, zq_0$  to H,

Hence the productions are renamed as:

$$S \rightarrow A \mid B$$

$$A \rightarrow bEA \mid bFC$$

$$B \rightarrow bEB \mid bFD$$

$$E \rightarrow bEE \mid bFH$$

$$F \rightarrow bEF \mid bFG$$

$$G \rightarrow \epsilon \mid aH$$

$$H \rightarrow aG$$

Q2. Give the equivalent CFG for the following PDA  $M = \{ \{q_0, q_1\}, \{0, 1\}, \{0, 1, z_0\}, \delta, q_0, z_0, \phi \}$  where  $\delta$  is defined by

$$\delta(q_0, \epsilon, z_0) = (q_1, \epsilon) - 1$$

$$\delta(q_0, 0, z_0) = (q_0, 0z_0) - 2$$

$$\delta(q_0, 0, 0) = (q_0, 00) - 3$$

$$\delta(q_0, 1, 0) = (q_0, 10) - 4$$

$$\delta(q_0, 1, 1) = (q_0, 11) - 5$$

$$\delta(q_0, 0, 1) = (q_1, \epsilon) - 6$$

$$\delta(q_1, 0, 1) = (q_1, \epsilon) - 7$$

$$\delta(q_1, 0, 0) = (q_1, \epsilon) - 8$$

$$\delta(q_1, \epsilon, z_0) = (q_1, \epsilon) - 9$$

Soln: First add the S transition

$$S \rightarrow [q_0, z_0, q_0] \mid [q_0, z_0, q_1]$$

Now consider ①

$$\delta(q_0, \epsilon, z_0) = (q_1, \epsilon)$$

$$[q_0, z_0, q_0] \rightarrow \epsilon$$

$$[q_0, z_0, q_1] \rightarrow \epsilon$$

$$\textcircled{2}, \quad \delta(q_0, 0, z_0) = (q_0, 0z_0)$$

$$[q_0, z_0, q_0] \rightarrow 0[q_0, 0, q_0][q_0, z_0, q_0]$$

$$[q_0, z_0, q_0] \rightarrow 0[q_0, 0, q_1][q_1, z_0, q_0]$$

$$[q_0, z_0, q_1] \rightarrow 0[q_0, 0, q_0][q_0, z_0, q_1]$$

$$[q_0, z_0, q_1] \rightarrow 0[q_0, 0, q_1][q_1, z_0, q_1]$$

$$\textcircled{3} \quad \delta(q_0, 0, 0) = (q_0, 00)$$

$$[q_0, 0, q_0] \rightarrow 0[q_0, 0, q_0][q_0, 0, q_0]$$

$$[q_0, 0, q_0] \rightarrow 0[q_0, 0, q_1][q_1, 0, q_0]$$

$$[q_0, 0, q_1] \rightarrow 0[q_0, 0, q_0][q_0, 0, q_1]$$

$$[q_0, 0, q_1] \rightarrow 0[q_0, 0, q_1][q_1, 0, q_1]$$

$$\textcircled{4} \quad \delta(q_0, 1, 0) = (q_0, 10)$$

$$[q_0, 1, q_0] \rightarrow 1[q_0, 1, q_0][q_0, 0, q_0]$$

$$[q_0, 1, q_0] \rightarrow 1[q_0, 1, q_1][q_1, 0, q_0]$$

$$[q_0, 1, q_1] \rightarrow 1[q_0, 1, q_0][q_0, 0, q_1]$$

$$[q_0, 1, q_1] \rightarrow 1[q_0, 1, q_1][q_1, 0, q_1]$$

$$\textcircled{5} \quad \delta(q_0, 1, 1) = (q_0, 11)$$

$$[q_0, 1, q_0] \rightarrow 1[q_0, 1, q_0][q_0, 1, q_0]$$

$$[q_0, 1, q_0] \rightarrow 1[q_0, 1, q_1][q_1, 1, q_0]$$

$$[q_0, 1, q_1] \rightarrow 1[q_0, 1, q_0][q_0, 1, q_1]$$

$$[q_0, 1, q_1] \rightarrow 1[q_0, 1, q_1][q_1, 1, q_1]$$

$$\textcircled{6} \quad \delta(q_0, 0, 1) = (q_1, \epsilon)$$

$$[q_0, 0, q_0] \rightarrow 0$$

$$[q_0, 0, q_1] \rightarrow 0$$

$$\textcircled{7} \quad \delta(q_1, 0, 1) = (q_1, \epsilon)$$

$$[q_1, q_1] \rightarrow 0$$

$$\textcircled{8} \quad \delta(q_1, 0, 0) = (q_1, \epsilon)$$

$$[q_1, 0, q_1] \rightarrow 0$$

$$\textcircled{9} \quad \delta(q_1, \epsilon, z_0) = (q_1, \epsilon)$$

$$[q_1, z_0, q_1] \rightarrow \epsilon$$

Now rename the variables :-

$q_0 z_0 q_0$  to A,  $q_0 z_0 q_1$  to B,  $q_0 0 q_0$  to C,

$q_0 0 q_1$  to D,  $q_1 z_0 q_0$  to E,  $q_1 z_0 q_1$  to F,

$q_1 0 q_0$  to G,  $q_1 0 q_1$  to H,  $q_0 1 q_0$  to I,

$q_0 1 q_1$  to J,  $q_1 1 q_0$  to K,  $q_1 1 q_1$  to L,

Hence the productions are renamed as :-

$$S \rightarrow A | B$$

$$A \rightarrow E | OCA | ODE$$

$$B \rightarrow E | OCB | ODF$$

$$C \rightarrow OCC | O DG | 1 IC | 1 JG$$

$$D \rightarrow OCD | O DH | 1 ID | 1 JH$$

$$E \rightarrow 1 II | 1 JK | 0$$

$$F \rightarrow 1 IJ | 1 JL | 0$$

$$G \rightarrow 0$$

$$H \rightarrow 0$$

$$I \rightarrow E$$

DPDA (Deterministic Push Down Automata)  
NPDA (Non-deterministic " " )

DPDA  
 $WCW^P$

We need central point  
because we need to  
know upto where to  
push & where to pop.

NPDA  
 $WLW^P$

No need of  
central point

accepts Deterministic  
CFL

DCFL (between regular  
& CFL)

7 tuples :-

Instantaneous description of PDA :-

ex: ababc bab' For given string write instantaneous  
description of PDA.

$$(q, ababc bab, \epsilon)$$

Consider 'a' and push into stack.

$$= (q_1, babcbab, a)$$

$$\text{push 'b'} = (q_1, abcbab, ba)$$

$$\text{push 'a'} = (q_1, bcbab, aba)$$

$$\text{push 'b'} = (q_1, cbab, bab)$$

while reading 'c' don't push 'c' into the  
stack as it is in "WCW" form.

~~Unit 5~~

Unit 5

$$= (q, bab, baba)$$

while reading 'b', the 'b' in stack will be popped.

$$= (q, ab, aba)$$

read 'a' & pop 'a' in the stack

$$= (q, b, ba)$$

read 'b' & pop 'b' in the stack.

$$= (q, \epsilon, a)$$

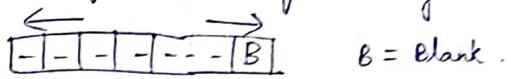
'a' is still present in the stack so it is not considerable.

Unit 5

17/7/21

Turing Machine :-

\* Discovered by Alan Turing in the year 1936.



\* If you are in a particular block we can go from left to right or from right to left.

7 tuples are,

$$(Q, \Sigma, \Gamma, \delta, q_0, B, F)$$

$Q$  = finite set of states

$\Sigma$  = finite input alphabet (never includes blank)

$\Gamma$  = tape alphabet (includes Blank)

$\delta$  = transition function

$$\delta = Q \times \Sigma \rightarrow Q \times \Gamma \times \{L, R\}$$

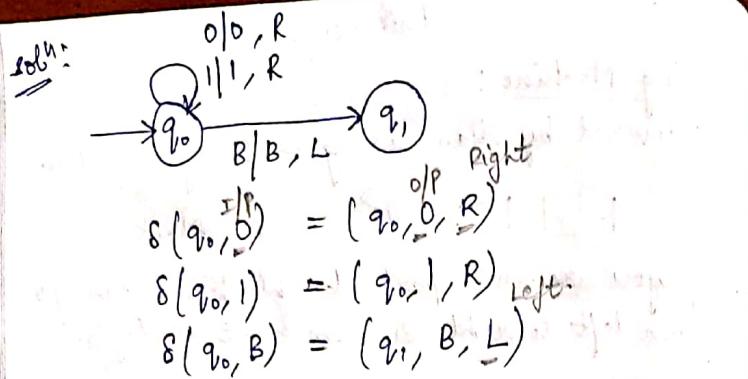
Depending on its present state & present tape alphabet, it will move to new state, change the tape symbol and move head pointer to either left or right.

$q_0$  - start state

$B$  - Blank symbol

$F$  - final state

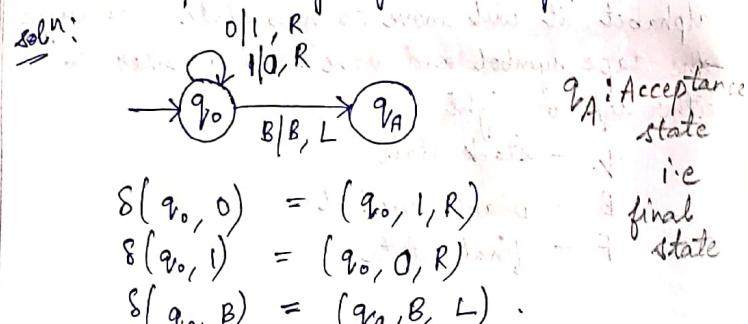
Q: Design a turing machine to accept the string belonging to the language  $(0+1)^*$ .



Tabular representation:

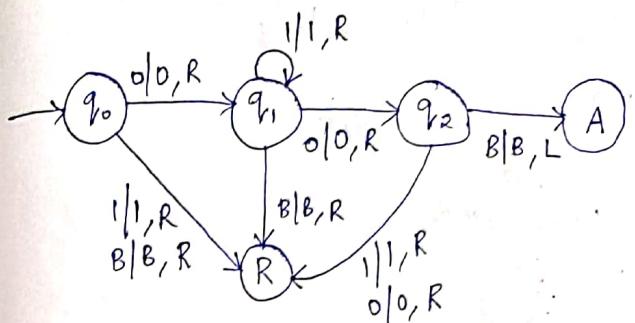
	0	1	B
$q_0$	$(q_0, 0, R)$	$(q_1, 1, R)$	$(q_1, B, L)$
$q_1$	-	-	-

H.W.  
Design a turing machine for finding 1's complement of a given binary number -



	0	1	B
$q_0$	$(q_0, 1, R)$	$(q_0, 0, R)$	$(q_A, B, L)$
$q_A$	-	-	-

Q1. Design a TM that accepts the string  $01^*0$ .  
Soln:  $00, 010, 0110, \dots$



$$\delta(q_0, 0) = (q_1, 0, R), \delta(q_0, 1) = (R, 1, R), \\ \delta(q_0, B) = (R, B, R)$$

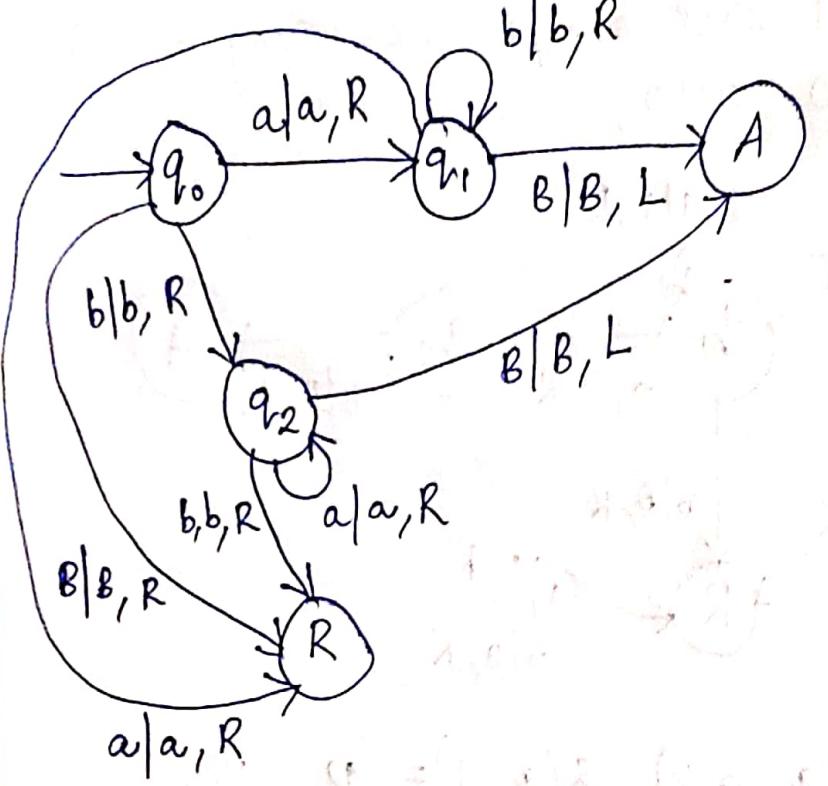
$$\delta(q_1, 0) = (q_2, 0, R), \delta(q_1, 1) = (q_1, 1, R), \delta(q_1, B) = (R, B, R)$$

$$\delta(q_2, 0) = (R, 0, R), \delta(q_2, 1) = (R, 1, R), \delta(q_2, B) = (A, B, L)$$

	0	1	B
$q_0$	$(q_1, 0, R)$	$(R, 1, R)$	$(R, B, R)$
$q_1$	$(q_2, 0, R)$	$(q_1, 1, R)$	$(R, B, R)$
$q_2$	$(R, 0, R)$	$(R, 1, R)$	$(A, B, L)$
A	-	-	-
R	-	-	-

Q2. Design a TM that accepts the strings  $ab^*$  or  $ba^*$ .

Soln:  $\{a, b, ab, abb, abbb, ba, baa, baaa, \dots\}$



$$\delta(q_0, a) = (q_1, a, R), \quad \delta(q_0, b) = (q_2, b, R), \quad \delta(q_0, B) = (R, B, R)$$

$$\delta(q_1, a) = (R, a, R), \quad \delta(q_1, b) = (q_2, b, R), \quad \delta(q_1, B) = (A, B, L)$$

$$\delta(q_2, a) = (q_1, a, R), \quad \delta(q_2, b) = (R, b, R), \quad \delta(q_2, B) = (A, B, L)$$

	a	b	B
$q_0$	$(q_1, a, R)$	$(q_2, b, R)$	$(R, B, R)$
$q_1$	$(R, a, R)$	$(q_2, b, R)$	$(A, B, L)$
$q_2$	$(q_1, a, R)$	$(R, b, R)$	$(A, B, L)$
A	—	—	—
R	—	—	—