PART-B

1. Convert the following NFA to equivalent DFA.

δ	0	1	ε
→A	B	B,D	–
B	–	D	E
C	E	–	–
*+ D	B,C	–	E
E	E	E	–

(i) converting NFA with ε to NFA without ε.

$$\delta_N(P,i) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(P),i))$$

ε-clos(A) = {A}
ε-clos(B) = {B,E}
ε-clos(C) = {C}
ε-clos(D) = {D,E}
ε-clos(E) = {E}

$\delta_N(A,0) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(A),0))$
$= \varepsilon\text{-clos}(\delta_\varepsilon(A,0))$
$= \varepsilon\text{-clos}(B)$
$= \{B,E\}$

$\delta_N(A,1) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(A),1))$
$= \varepsilon\text{-clos}(\delta_\varepsilon(A,1))$
$= \varepsilon\text{-clos}(B,D)$
$= \{B,D,E\}$

$\delta_N(B,0) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(B),0))$
$= \varepsilon\text{-clos}(\delta_\varepsilon(B,0) \cup \delta_\varepsilon(E,0))$
$= \varepsilon\text{-clos}(\phi \cup E)$
$= \{E\}$

$\delta_N(B,1) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(B),1))$
$= \varepsilon\text{-clos}(\delta_\varepsilon(B,1) \cup \delta_\varepsilon(E,1))$
$= \varepsilon\text{-clos}(D \cup E)$
$= \{D,E\}$

$\delta_N(C,0) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(C),0))$

$= \varepsilon\text{-clos}(\delta_\varepsilon(C,0))$

$= \varepsilon\text{-clos}(E)$

$= \{E\}$

$\delta_N(C,1) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(C),1))$

$= \varepsilon\text{-clos}(\delta_\varepsilon(C,1))$

$= \varepsilon\text{-clos}(\phi)$

$= \phi$

$\delta_N(D,0) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(D),0))$

$= \varepsilon\text{-clos}(\delta_\varepsilon(D,0) \cup \delta_\varepsilon(E,0))$

$= \varepsilon\text{-clos}(B \cup E)$

$= \{B,C,E\}$

$\delta_N(D,1) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(D),1))$

$= \varepsilon\text{-clos}(\delta_\varepsilon(D,1) \cup \delta_\varepsilon(E,1))$

$= \varepsilon\text{-clos}(\phi \cup E)$

$= \{E\}$

$\delta_N(E,0) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(E),0))$

$= \varepsilon\text{-clos}(\delta_\varepsilon(E,0))$

$= \varepsilon\text{-clos}(E)$

$= \{E\}$

$\delta_N(E,1) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(E),1))$

$= \varepsilon\text{-clos}(\delta_\varepsilon(E,1))$

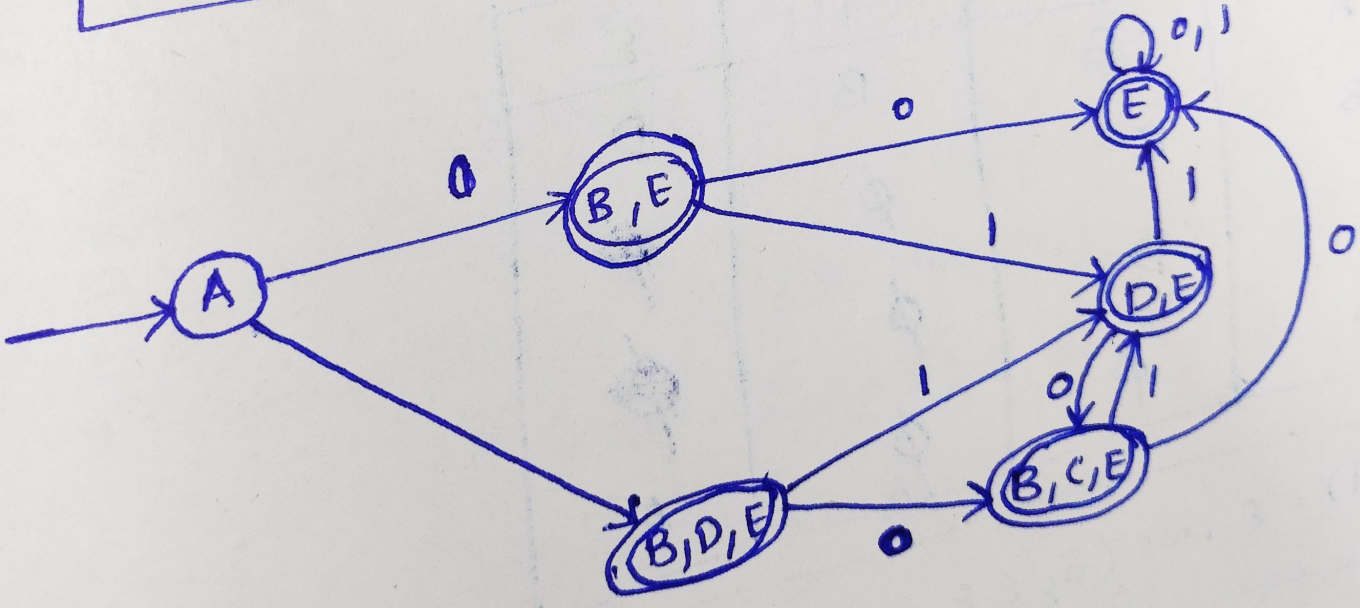
$= \varepsilon\text{-clos}(E)$

$= \{E\}$

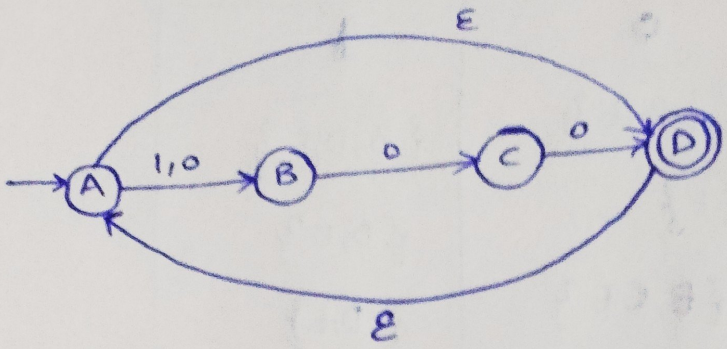
δ	0	1
$\rightarrow$ A	$\{B,E\}$	$\{B,D,E\}$
* B	$\{E\}$	$\{D,E\}$
C	$\{E\}$	ϕ
* D	$\{B,C,E\}$	$\{E\}$
* E	$\{E\}$	$\{E\}$

(ii) NFA to DFA conversion:

δ_D	0	1
→ {A}	{B,E}	{B,D,E}
* {B,E}	{E}	{D,E}
* {B,D,E}	{B,C,E}	{D,E}
* {E}	{E}	{E}
* {D,E}	{B,C,E}	{E}
* {B,C,E}	{E}	{D,E}



2. Construct a NFA for the given NFA with ε moves.



ε-clos(A) = {A, D}
ε-clos(B) = {B}
ε-clos(C) = {C}
ε-clos(D) = {D, A}

δ_ε	0	1	ε
A	B	B	D
B	C	ϕ	
C	D	ϕ	
D	ϕ	ϕ	A

$$\delta_N(P, i) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(P), i))$$

$$\delta_N(A, 0) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(A), 0))$$
$$= \varepsilon\text{-clos}(\delta_\varepsilon(A, 0) \cup \delta_\varepsilon(D, 0))$$
$$= \varepsilon\text{-clos}(B \cup \phi)$$
$$= \{B\}$$

$$\delta_N(A, 1) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(A), 1))$$
$$= \varepsilon\text{-clos}(\delta_\varepsilon(A, 1) \cup \delta_\varepsilon(D, 1))$$
$$= \varepsilon\text{-clos}(B \cup \phi)$$
$$= \{B\}$$

$$\delta_N(B, 0) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(B), 0))$$
$$= \varepsilon\text{-clos}(\delta_\varepsilon(B, 0))$$
$$= \varepsilon\text{-clos}(C)$$

$\delta_N(B,1) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(B),1))$

$= \varepsilon\text{-clos}(\delta_\varepsilon((B),1))$

$= \varepsilon\text{-clos}(\phi)$

$= \phi$

$\delta_N(C,0) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(C),0))$

$= \varepsilon\text{-clos}(\delta_\varepsilon(C,0))$

$= \varepsilon\text{-clos}(D)$

$= \{A,D\}$

$\delta_N(C,1) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(C),1))$

$= \varepsilon\text{-clos}(\delta_\varepsilon(C,1))$

$= \varepsilon\text{-clos}(\phi)$

$= \phi$

$\delta_N(D,0) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(D),0))$

$= \varepsilon\text{-clos}(\delta_\varepsilon(A,D) \cup \delta_\varepsilon(D,0))$

$= \varepsilon\text{-clos}(B \cup \phi)$

$= \{B\}$

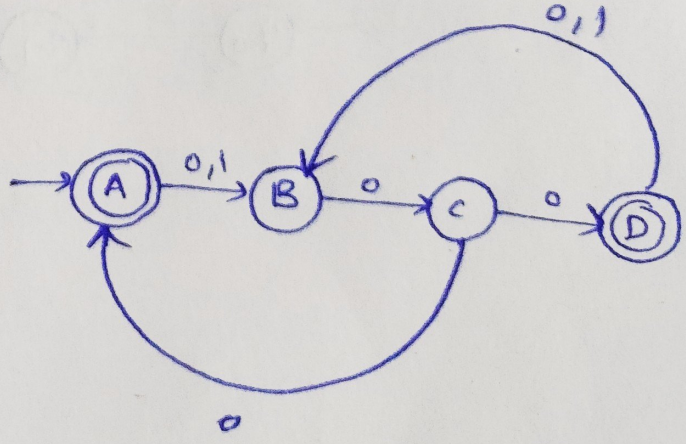
$\delta_N(D,1) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(D),1))$

$= \varepsilon\text{-clos}(\delta_\varepsilon(A,1) \cup \delta_\varepsilon(D,1))$

$= \varepsilon\text{-clos}(B,\phi)$

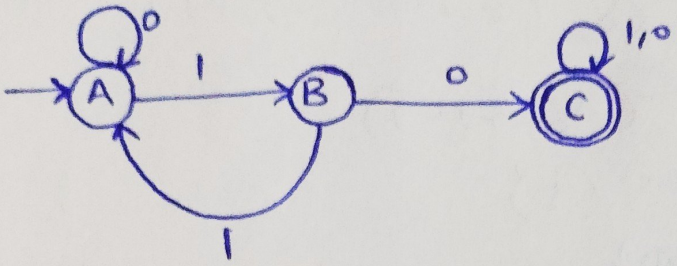
$= \{B\}$

	0	1
→*A	B	B
B	C	ϕ
C	A,D	ϕ
D	B	B



3. Draw the transition diagram for the DFA accepting all strings with a substring 10.

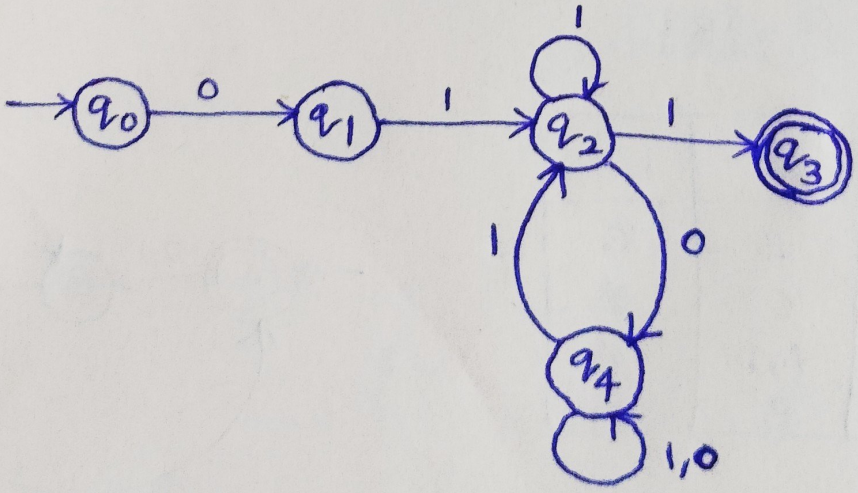
Regular Expression: $(0+1)^* 10 (0+1)^*$



δ_D	0	1
→A	A	B
B	C	A
*C	C	C

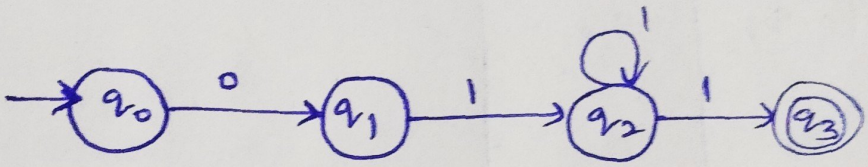
4. Design a NFA accept the following string over the alphabet (0,1). The set of all string that begin with 01 and end with 11. Check for the validity of 01111 and 0110 string.

Regular Expression: $011 + 01 (0+1)^* 11$



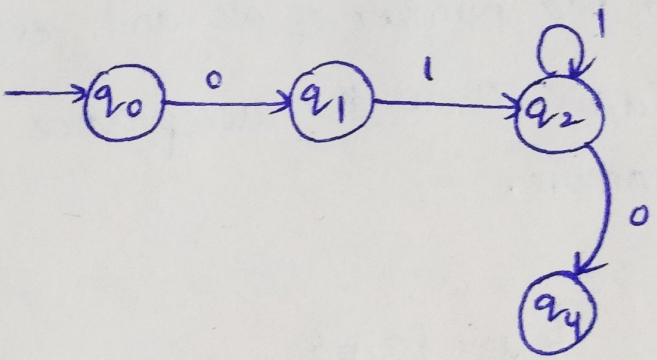
δ_N	0	1
$\rightarrow q_0$	q_1	–
q_1	–	q_2
q_2	q_4	q_2, q_3
$*q_3$	–	–
q_4	q_4	q_2, q_4

(i) 0 1 1 1 1



01111 reached the final state and
01111 starts with 01 & ends with 11.

(ii) 0110

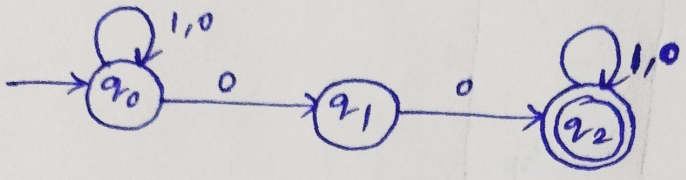


0110 doesn't reach the final state and
0110 starts with 01 & ends with 10
as it is invalid string.

∴ It is a Valid NFA machine.

5. Construct an NFA accepting binary string with two consecutive 0's.

Regular Expression: $(0+1)^*00(0+1)^*$



δ_N	0	1
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	–
$*\ q_2$	q_2	q_2

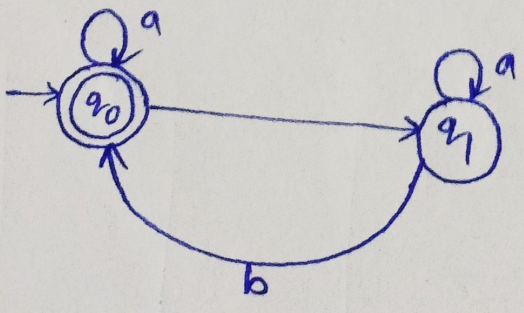
6. Design DFA to accept odd number of a's and even number of b's, where $\Sigma = \{a,b\}$. Show the acceptance of a string with an example.

$\Rightarrow n_a(w) = \text{odd}$
$\Rightarrow n_b(w) = \text{even}$
$\}\ w = \{a,b\}$

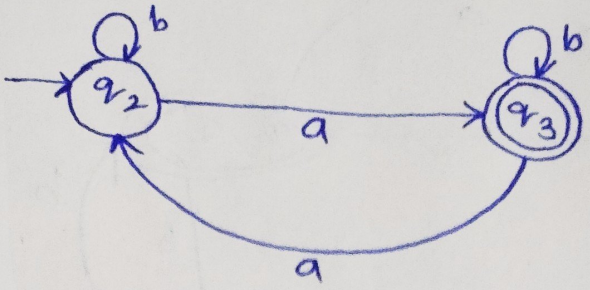
$FA_a\ (n_a) = b^*ab^*(b^*ab^*ab^*)^*$

$FA_b\ (n_b) = a^*(a^*ba^*ba^*)^*a^*$

FA_a }



FA_b }



$FA_a = \{q_0, q_1\}$

$FA_b = \{q_2, q_3\}$

$FA_a \times FA_b = \{(q_0,q_2), (q_0,q_3), (q_1,q_2), (q_1,q_3)\}$

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

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

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

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

$\delta((q_0,q_2),a) = \delta(q_0,a) \cup \delta(q_2,a) = \{q_0, q_3\}$

$\delta((q_0,q_2),b) = \delta(q_0,b) \cup \delta(q_2,b) = \{q_1, q_2\}$

$\delta((q_0,q_3),a) = \delta(q_0,a) \cup \delta(q_3,a) = \{q_1, q_3\}$

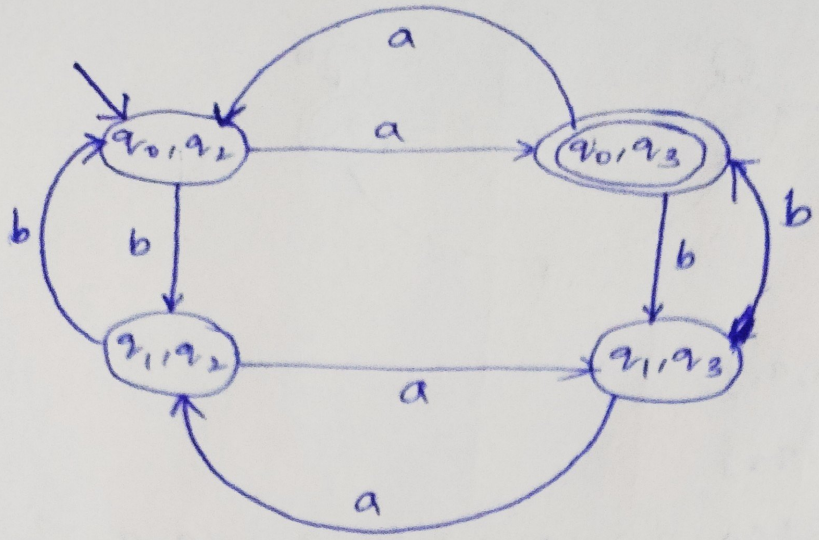
$\delta((q_0,q_3),b) = \delta(q_0,b) \cup \delta(q_3,b) = \{q_0, q_2\}$

$\delta((q_1,q_2),a) = \delta(q_1,a) \cup \delta(q_2,a) = \{q_1, q_3\}$

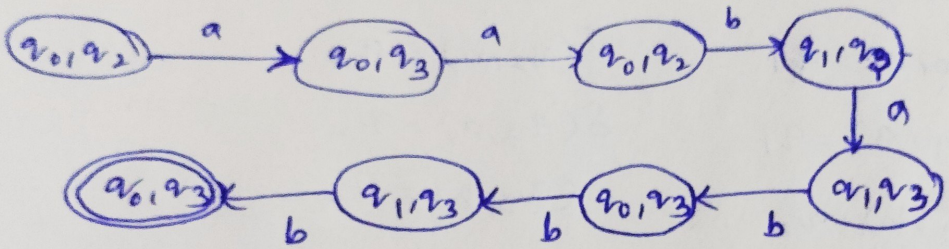
$\delta((q_1,q_2),b) = \delta(q_1,b) \cup \delta(q_2,b) = \{q_1, q_3\}$

$\delta((q_1,q_3),a) = \delta(q_1,a) \cup \delta(q_3,a) = \{q_0, q_2\}$

$\delta((q_1,q_3),b) = \delta(q_1,b) \cup \delta(q_3,b) = \{q_0, q_3\}$



Ex: aababbb

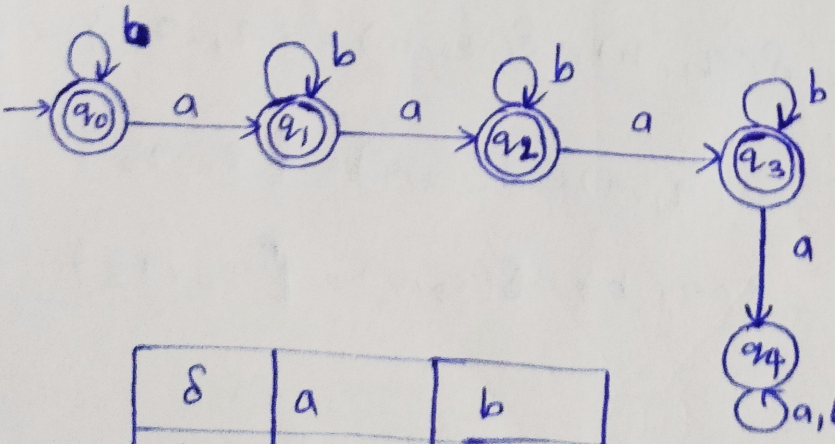


∴ It reached final state.
So the machine is valid.

7. Design DFA for the following over (a,b)

(i) All string containing not more than three a's.

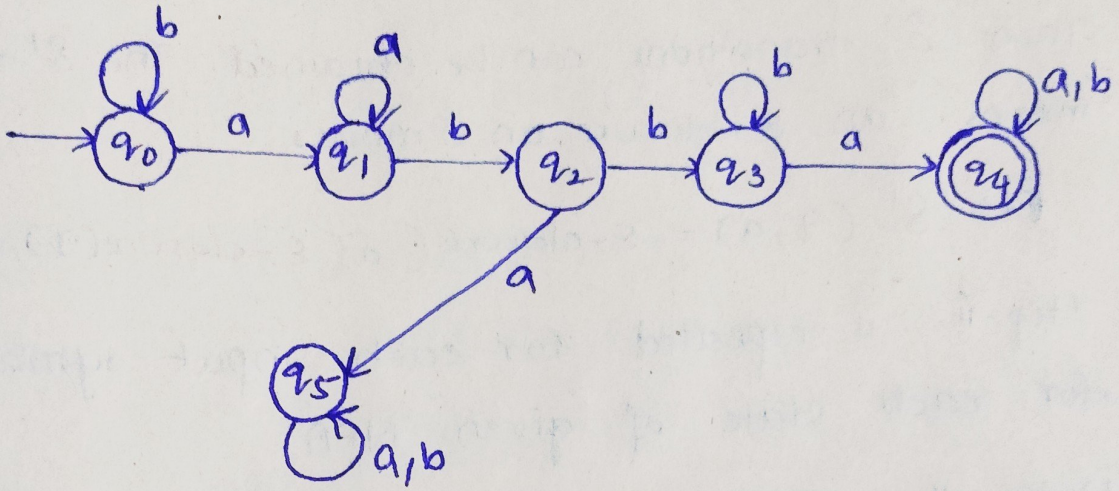
Regular Expression: $b^*+b^*ab^*+b^*aab^*+b^*aaab^*$



δ	a	b
→*q_0	q_1	q_0
*q_1	q_2	q_1
*q_2	q_3	q_2
*q_3	q_4	q_3
q_4	q_4	q_4

(ii) All strings that has at least two occurences of b between any two occurence of a.

$(a+b)^* a\, bb^+ a\, (a+b)^*$



δ	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
q_2	q_5	q_3
q_3	q_4	q_3
$*\ q_4$	q_4	q_4
q_5	q_5	q_5

8. Explain procedure for converting NFA with ε moves to without ε with suitable Example.

Steps:

(i) Find out all the ε transitions from each state from Q that will be called as ε-closure $\{q_i\}$ where $q_i \in Q$.

(ii) Then δ' transitions can be obtained. The δ' transitions means an ε-closure on δ moves.

Ex: $\delta'(q,a) = \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(q), a))$

(iii) Step-ii is repeated for each input symbol and for each state of given NFA.

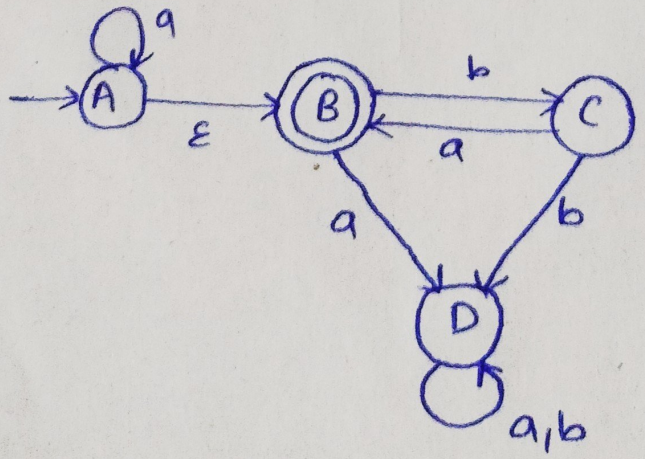
(iv) Using the resultant states the transition table for equivalent NFA without ε can be built.

(v) $F' = F \cup \{q_0\}$ if ε-closure(q_0) is having F as a member

F otherwise

Ex: Go to problem (2).

9. Convert following NFA with epsilon to DFA.



(i) Converting ε-NFA to NFA.

$\varepsilon\text{-clos}(A) = \{A, B\}$

$\varepsilon\text{-clos}(B) = \{B\}$

$\varepsilon\text{-clos}(C) = \{C\}$

$\varepsilon\text{-clos}(D) = \{D\}$

$$\delta_N(P_i, i) = \varepsilon\text{-clos}(\delta_\varepsilon(\varepsilon\text{-clos}(P), i))$$

$$\delta_N(A,a) = \varepsilon\text{-clos}(\delta_\varepsilon((A,B),a))$$
$$= \varepsilon\text{-clos}(\delta_\varepsilon(A,a) \cup \delta_\varepsilon(B,a))$$
$$= \varepsilon\text{-clos}(A,D)$$
$$= \{A,B,D\}$$

$$\delta_N(A,b) = \varepsilon\text{-clos}(\delta_\varepsilon((A,B),b))$$
$$= \varepsilon\text{-clos}(\delta_\varepsilon(A,b) \cup \delta_\varepsilon(B,b))$$
$$= \varepsilon\text{-clos}(C)$$
$$= \{C\}$$

$$\delta_N(B,a) = \varepsilon\text{-clos}(\delta_\varepsilon(B,a))$$
$$= \varepsilon\text{-clos}(D)$$
$$= \{D\}$$

$$\delta_N(B,b) = \varepsilon\text{-clos}(\delta_\varepsilon(B,b))$$
$$= \varepsilon\text{-clos}(C)$$
$$= \{C\}$$

$$\delta_N(C,a) = \varepsilon\text{-clos}(\delta_\varepsilon(C,a))$$
$$= \varepsilon\text{-clos}(B)$$
$$= \{B\}$$

$$\delta_N(C,b) = \varepsilon\text{-clos}(\delta_\varepsilon(C,b))$$
$$= \varepsilon\text{-clos}(D)$$
$$= \{D\}$$

$$\delta_N(D,a) = \varepsilon\text{-clos}(\delta_\varepsilon(D,a))$$

$$= \varepsilon\text{-clos}(D)$$

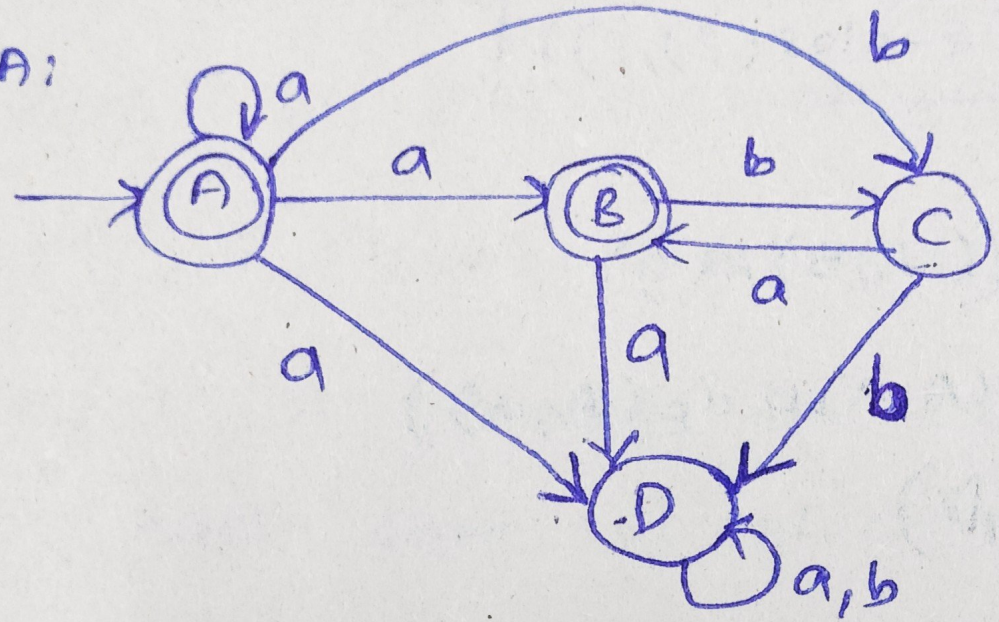
$$= \{D\}$$

$$\delta_N(D,b) = \varepsilon\text{-clos}(\delta_\varepsilon(D,b))$$

$$= \varepsilon\text{-clos}(D)$$

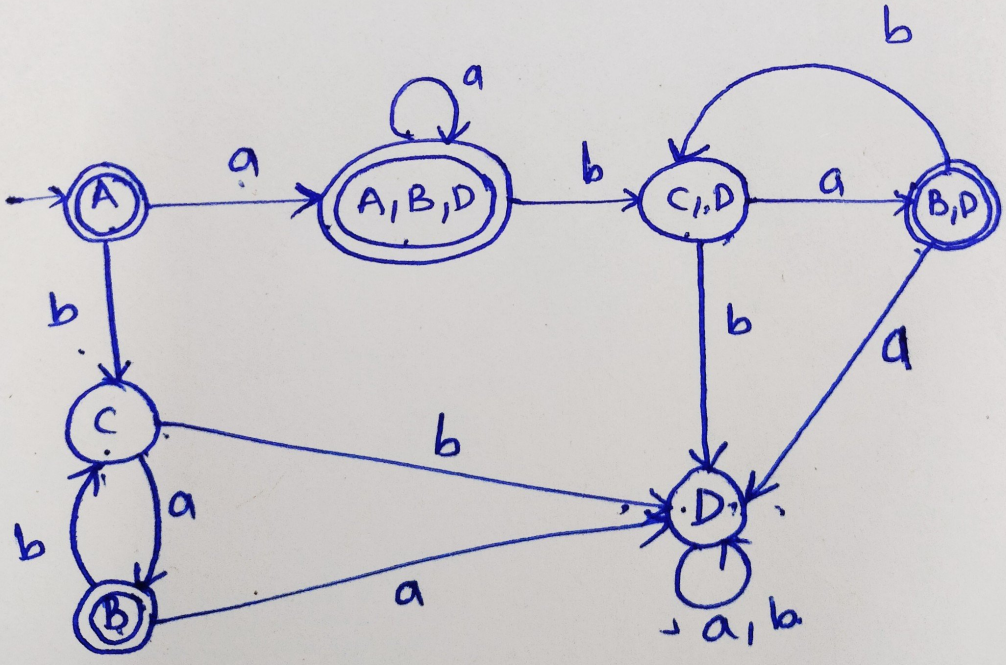
$$= \{D\}$$

NFA:



(ii) Converting NFA to DFA:

δ_D	a	b
$\rightarrow$* {A}	{A,B,D}	{C}
* {A,B,D}	{A,B,D}	{C,D}
{C}	{B}	{D}
{C,D}	{B,D}	{D}
* {B}	{D}	{C}
{D}	{D}	{C}
* {B,D}	{D}	{C,D}



10. State and prove Arden's theorem.

If P and Q are 2 Regular Expressions over Σ, and if P does not contain ε, then the following equation in R given by $R = Q + RP$ has a unique solution, i.e., $R = QP^*$

$R = Q + RP$

$= Q + QP^*P$ $\{R = QP^*\}$

$= Q(\varepsilon + P^*P)$ $\{\varepsilon + R^*R = R^*\}$

$= QP^*$

$R = Q + RP$

$= Q + [Q + RP]P$

$= Q + QP + RP^2$

$= Q + QP + [Q + RP]P^2$

$= Q + QP + QP^2 + QP^3$

$\vdots$

$= Q + QP + QP^2 + \cdots + QP^n + RP^{n+1}$ $\{R = QP^*\}$

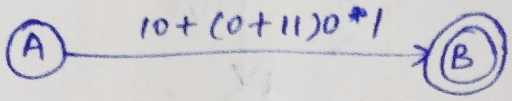
$= Q + QP + QP^2 + \cdots + QP^n + QP^*P^{n+1}$

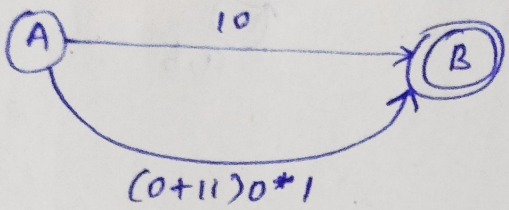
$= Q[\varepsilon + P + P^2 + \cdots + P^n + P^*P^{n+1}]$

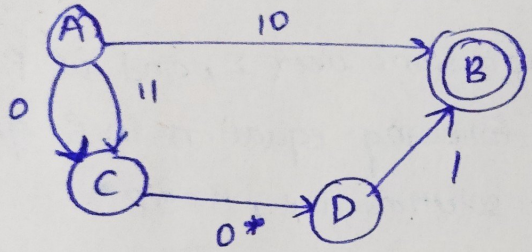
$= Q[P^*] \Rightarrow QP^*$

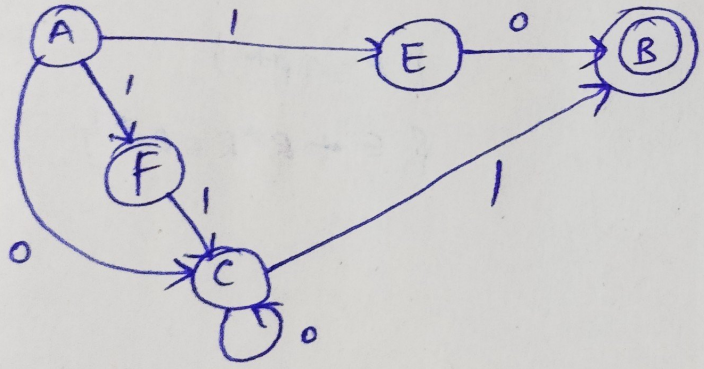
11. Construct DFA with reduced states equivalent to regular expression.

$10+(0+11)0^*1$.





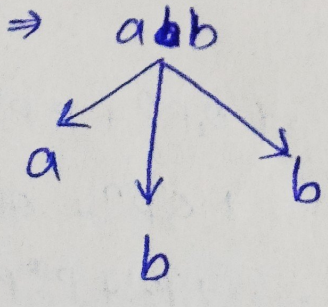




12. Prove $L=\{a^n b^{2n} \mid n>=0\}$ is not regular language using pumping lemma.

$L=\{a^n b^{2n} \mid n \geq 0\}$

Ex: ε, abb, aabbbb, aaabbbbbb.



~~$a(abb)^i b \in L$~~

$ab^i b \in L$

$\Rightarrow i = 0$

$ab \notin L$

$abb \in L$

$abbb \notin L$

} not regular.

13- Construct Minimum state automata for the following.

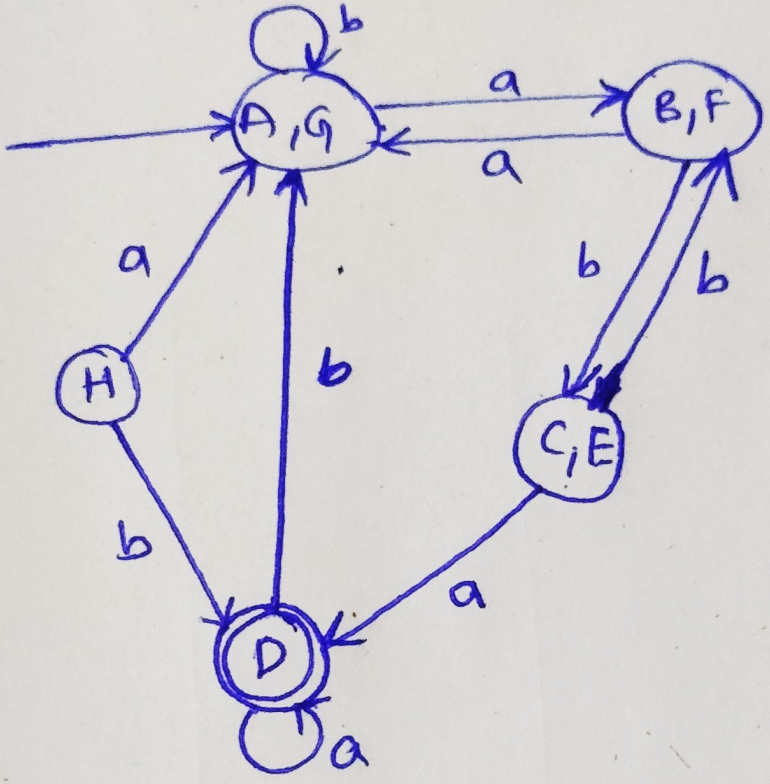
	a	b
→ A	B	A
B	A	C
C	D	B
D (final)	D	A
E	D	F
F	G	E
G	F	G
H	G	D

1- equivalence:

{D} {A,B,F,G} {C,E} {H}

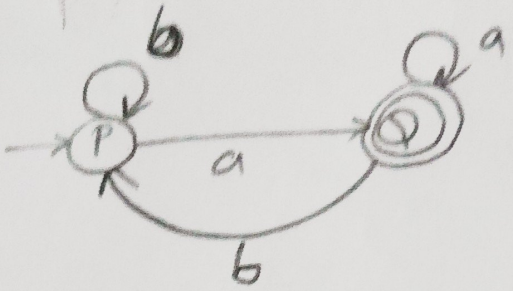
2. equivalence:

{D} {A,G} {B,F} {C,E} {H}



14.

	a	b
→ P	Q	P
Q*	Q	P



$P = \varepsilon + P.b + Q.b$ — (1)

$Q = Q.b + P.a$ — (2).

from (1),

$P = P.b + \underbrace{Q.b + \varepsilon}_{Q}$

$P = (Qb + \varepsilon)b^*$

$P = Qbb^* + b^*$

from (2),

$Q = Q.b + P.a$

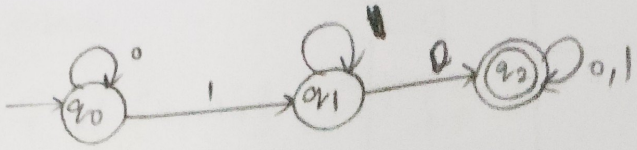
$Q = Q.b + Qbb^*a + b^*a$

$Q = Q(\underbrace{b + bb^*a}_{P}) + b^*a$

$Q = b^*a(b + bb^*a)^*$

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

15.



$q_0 = q_0 \cdot 0 + \varepsilon$

$q_1 = q_0 \cdot 1 + q_1 \cdot 1$

$q_2 = q_1 \cdot 0 + q_2 \cdot 0 + q_2 \cdot 1$

$q_2 = q_1 \cdot 0 + q_2 (0+1)$

since q_2 is final state, so the resultant is q_2

$$q_2 = q_1 \cdot 0 + q_2 (0+1) \quad \text{—(1)}$$

$q_0 = q_0 \cdot 0 + \varepsilon$

$R = R \cdot P + Q$ (from Arden's method).

$R = QP^*$

$\Rightarrow \varepsilon \cdot 0^*$

$\Rightarrow \boxed{q_0 = 0^*}$ —(2)

$q_1 = q_0 \cdot 1 + q_1 \cdot 1$

$\Rightarrow q_1 = 0^* \cdot 1 + q_1 \cdot 1$

$R = Q \quad + \quad R \cdot P$

$R = QP^*$

$\Rightarrow 0^* \cdot 1 \cdot 1^* \quad (1 \cdot 1^* = 1^+)$

$\boxed{q_1 = 0^* \cdot 1^+}$ —(3)

From (1) to (3),

$q_2 = q_1 \cdot 0 + q_2 (0+1)$

$q_2 = 0^* \cdot 1^+ \cdot 0 + q_2 (0+1)$

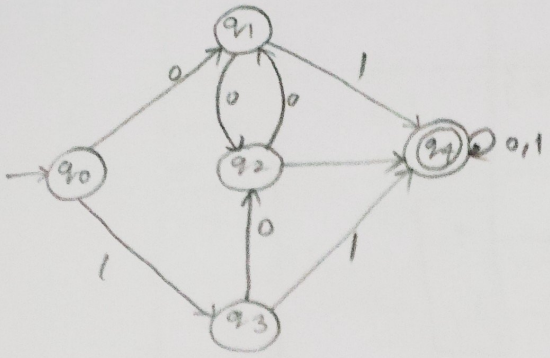
$R = \quad Q \quad R \quad P$

$R = QP^*$

$\Rightarrow q_2 = 0^* \cdot 1^+ \cdot 0 \cdot (0+1)^*$

∴ Regular Expression: $0^* 1^+ 0 (0+1)^*$

16.



minimize DFA.

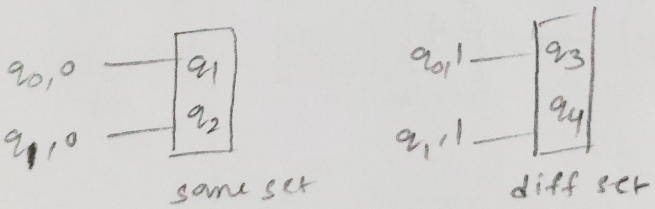
	0	1
q_0	q_1	q_3
q_1	q_2	q_4
q_2	q_4	q_4
q_3	q_2	q_4
q_4	q_4	q_4

0-equivalence:

$\{q_0, q_1, q_2, q_3\}$ $\{q_4\}$

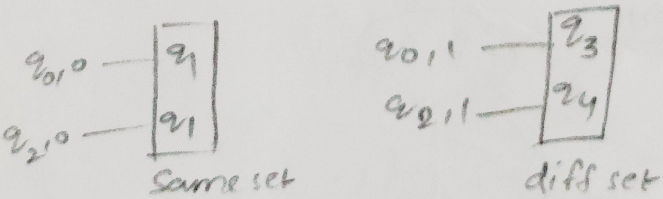
1-equivalence: (q_0 with q_1)

$\{q_0\}$ $\{q_1, q_2, q_3\}$ $\{q_4\}$



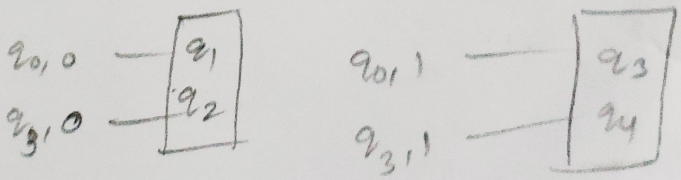
$\therefore q_0 \neq q_1$

(q_0 with q_2)



$\therefore q_0 \neq q_2$

(q_0 with q_3)

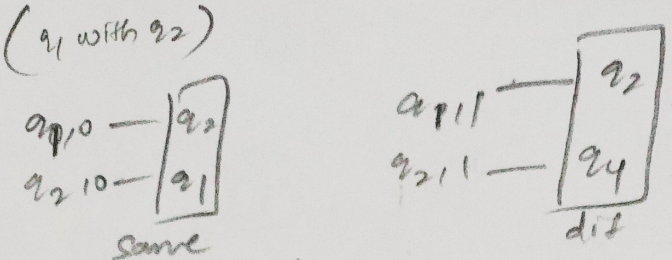


$\therefore q_0 \neq q_3$

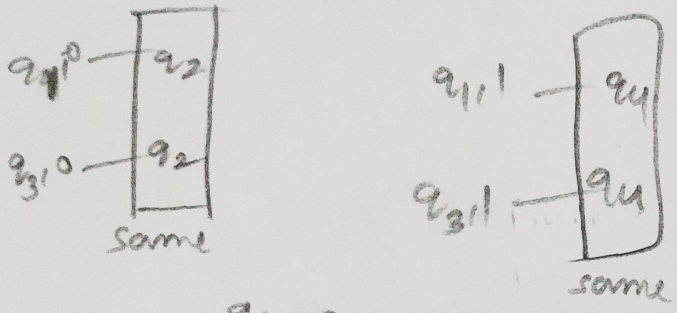
2 - equivalence:

$\{q_0\}\ \{q_1, q_3\}\ \{q_2\}\ \{q_4\}$

(q_1 with q_2)



$q_1 \neq q_2$

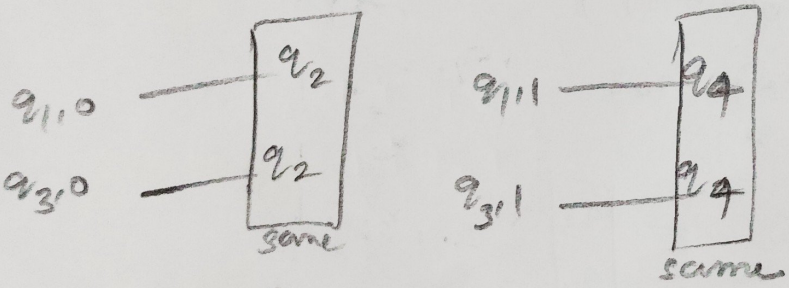


$q_1 = q_3$

3 - equivalence:

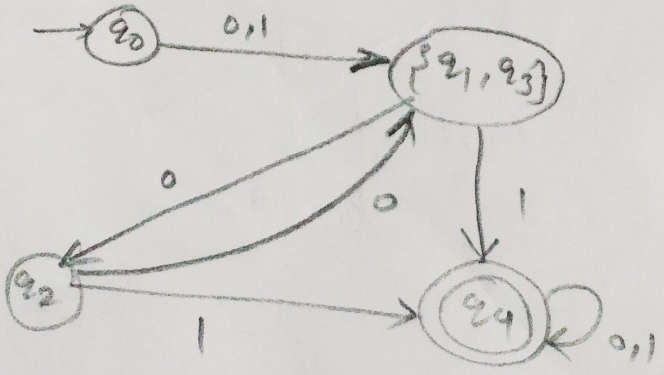
$\{q_0\}\ \{q_1, q_3\}\ \{q_2\}\ \{q_4\}$

(q_1 with q_3)



$q_1 = q_3$

∴ Resultant $\}$ $\{q_0\}\ \{q_1 q_3\}\ \{q_2\}\ \{q_4\}$



4. Explain about pumping Lemma Theorem and Determine whether the following language are regular or not with proper justification.

$$L = \{ a^n b^n / n \geq 1 \}$$

Pumping Lemma Theorem:

* Pumping Lemma is used to prove that a language is NOT Regular.
* The Pumping lemma states that for any regular language, there exists a length such that any string longer than its length can be divided into three parts, and by repeating or removing the middle part, the resulting string will also be in the language.
* The pumping Lemma can be formally stated as follows –
If L is a regular language, then there exists an integer n (the pumping length) such that any string w in L with $|w| \geq n$ can be decomposed into three parts, $w = xyz$, satisfying the following conditions —

$|xy| \leq n |xy|$,

$|y| > 0$,

$x y^i z \in L$ for all $i \geq 0$.

* These conditions states that for any sufficiently long string in a regular language, there is a section of the string that can be "pumped" (repeated or removed) to produce new string that also belong to the language.

To prove that a language is not Regular using pumping lemma, follow the below steps:

(i) Assume that A is regular.

(ii) It has to have a pumping length (say P).

(iii) All strings longer than P can be pumped $|w| \geq P$.

(iv) Now find a string 'w' in A such that $|w| \geq P$.

(v) Divide w into x y z.

(vi) Show that $xy^iz \notin L$ for some i.

(vii) Then consider all ways that w can be divided into x y z.

(viii) Show that none of these can satisfy all 3 pumping conditions at the same time.

(ix) Therefore L is not a regular language. (w cannot be pumped = contradiction)

* $L = \{a^n b^n \mid n \geq 1\}$

Let L be a Regular language.

$L = \{ab, aabb, aaabbb, \ldots\}$

let $n = 4$, $w = aabb$, $|w| = 4 \geq 4$

$\Rightarrow \underline{a}\,\underline{a}\,\underline{b}\,\underline{b}$

x y z

$\Rightarrow |xy| \leq n$

$\Rightarrow |a.ab| \leq 4$

$\Rightarrow 3 \leq 4$ (✓).

$\Rightarrow |y| > 0$

$\Rightarrow |ab| > 0$

$\Rightarrow 2 > 0$. (✓)

$\Rightarrow xy^iz \in L$

$\rightarrow a(ab)^i b.$

for $i=0$, $ab \in L$

for $i=1$, $aabb \in L$

for $i=2$, $a(ab)^2b \Rightarrow aabab b \notin L$

$\therefore$ Contradiction.

$\because$ $L = \{a^n b^n \mid n \geq 1\}$ is not a regular language.

18.

	C	D
→ * q_1	q_1	q_2
q_2	q_3	q_1
q_3	q_2	q_3

	C	D
→ * q_4	q_4	q_5
q_5	q_6	q_4
q_6	q_7	q_6
q_7	q_6	q_4

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

2 machines are Equivalent

5. Show that the following grammar is ambiguous.

$E \rightarrow E+E \mid E-E \mid E*E \mid E/E \mid (E) \mid a$ where E is the start symbol.

$E \rightarrow E+E$

$E \rightarrow E-E$

$E \rightarrow E*E$

$E \rightarrow E/E$

$E \rightarrow (E)$

$E \rightarrow a$

$V = \{E\}$

$T = \{+, -, *, /, a\}$

An ambiguous grammar is a context free grammar for which there exists a string that can have more than one leftmost derivation or parse tree. This means that a string can be generated in multiple ways, leading to different meanings.

let us consider string $a+a*a$.

LMD (+)

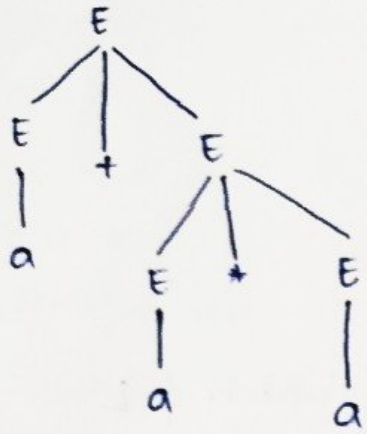
$E \rightarrow E+E$

$\rightarrow a+E$

$\rightarrow a+E*E$

$\rightarrow a+a*E$

$\rightarrow a+a*a$



RMD (+)

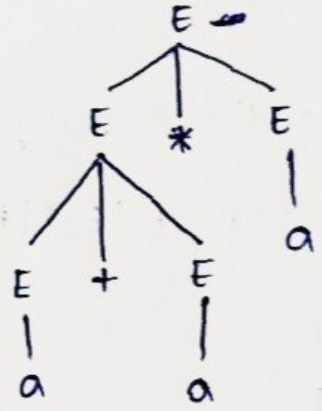
$E \rightarrow E+E$

$E \rightarrow E+E*E$

$E \rightarrow E+E*a$

$\rightarrow E+a*a$

$\rightarrow a+a*a$



LMD (*)

$E \rightarrow E*E$

$\rightarrow E+E*E$

$\rightarrow a+E*E$

$\rightarrow a+a*E$

$\rightarrow a+a*a$

RMD(*)

$E \rightarrow E*E$

$\rightarrow E*a$

$\rightarrow E+E*a$

$\rightarrow E+a*a$

$\rightarrow a+a*a$

∴ Both parse trees are derived from the same grammar rules but both parse trees are different. Hence the grammar is Ambiguous.

21. Write CFG for the language $\{a^n b^n \mid n \geq 1\}$

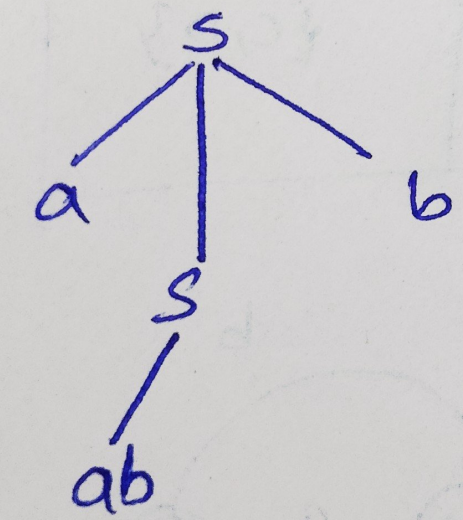
$L = \{a^n b^n \mid n \geq 1\}$

$L = \{ab, a^2b^2, a^3b^3, \ldots\}$

$S \rightarrow aSb \mid ab$

$S \rightarrow aSb$

$S \rightarrow a.ab.b$



$G = (V, \Sigma, R, S)$

$= (\{S, a, b\}, \{a, b\}, R, S\}$

where

$R = \{S \rightarrow asb \mid ab\}$