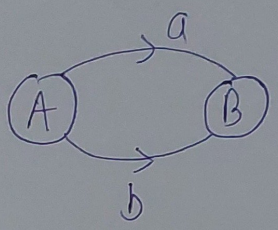
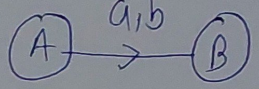
→ Conversion of Regular Expression to Finite Automata :→ ①

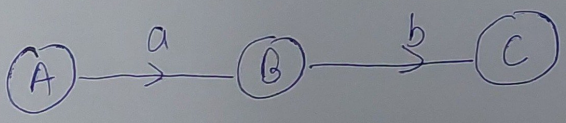
There are certain rules that, we need to follow :→

1> If (a+b) is given, then automata can be formed like :→

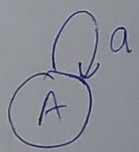
(a+b) → 

or 

2> If (a.b) is given, then automata can be formed like :→



3> If (a*) is given then automata can be formed like :→

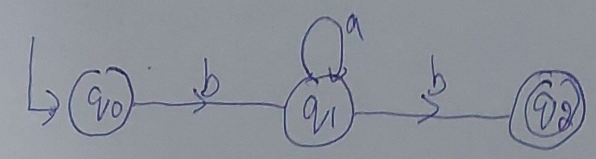


Keeping these three rules in mind, we can convert any regular expression to an automata.

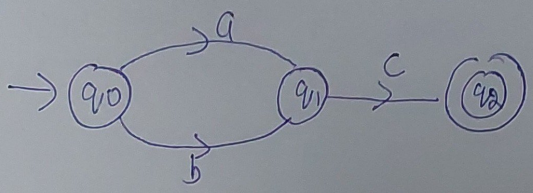
P.T.O

Convert the following Regular Expressions to their equivalent Finite Automata :-

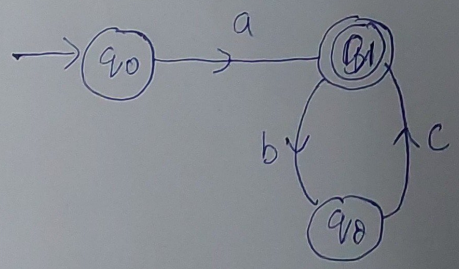
1> ba^*b



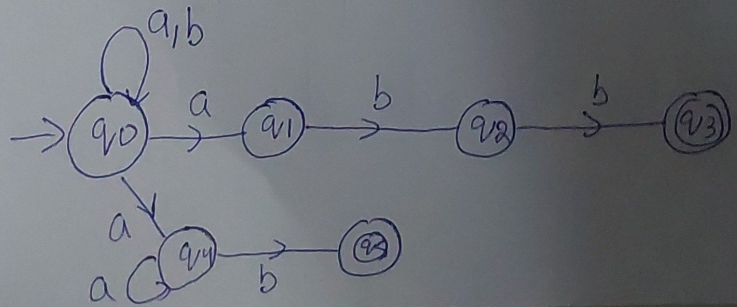
2> $(a+b)c$



3> $a(bc)^*$



4> $(a|b)^*(abb|a^*b)$ Here "|" is equivalent to "+".

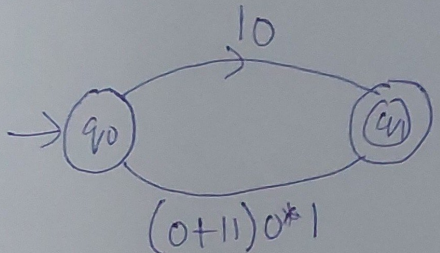


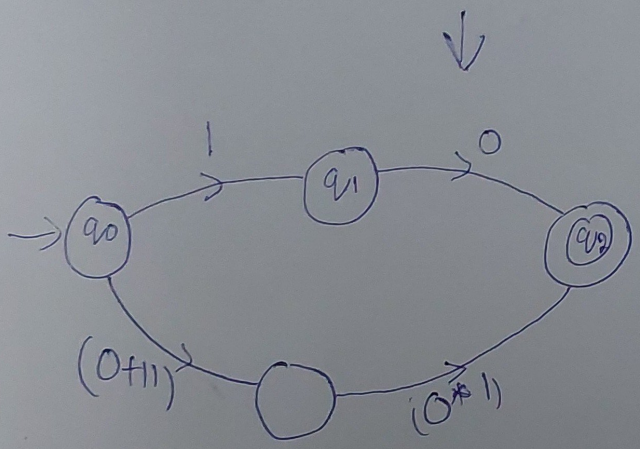
(4c)

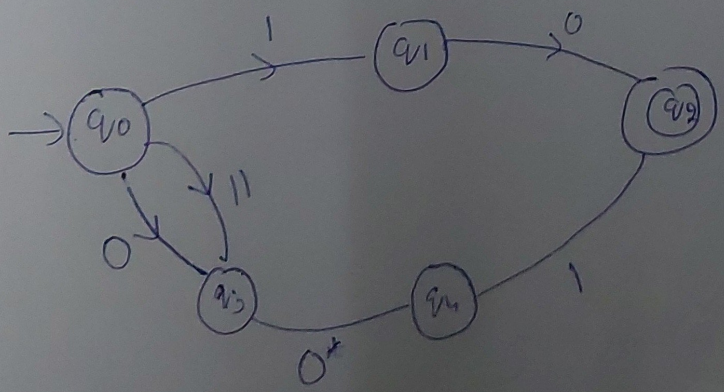
5) Convert the following Regular Expression to its equivalent finite automata. :- (3)

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

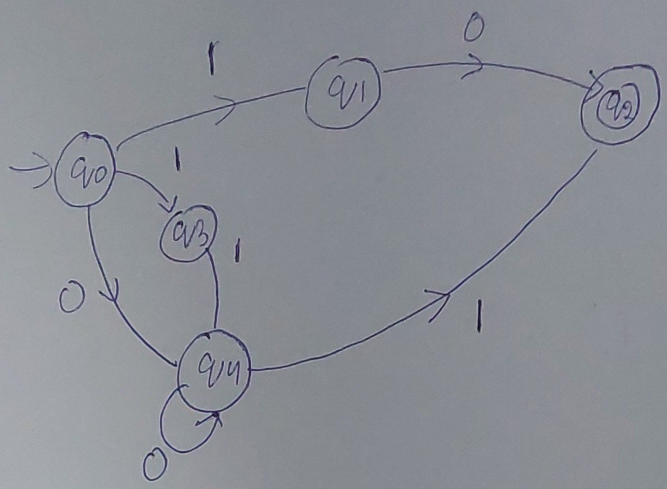
Let's do step by step :-





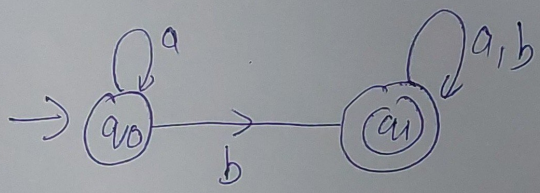


P.T.O

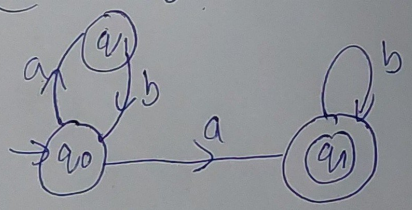


—o—

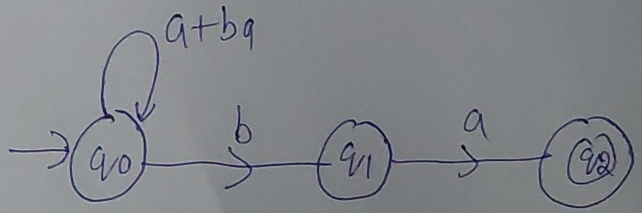
6 → $a^*b(a+b)^*$



7 → $(ab)^*ab^*$

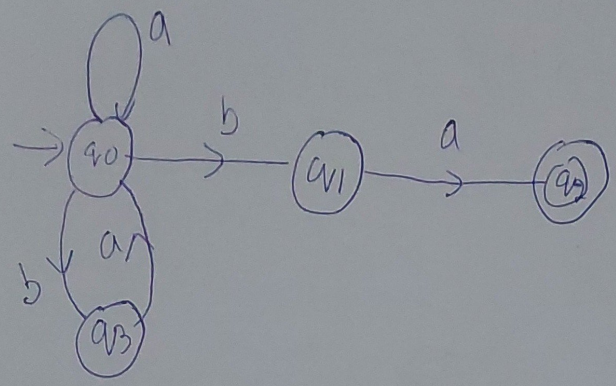


8) $(a+ba)^*ba$

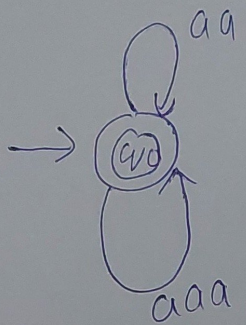


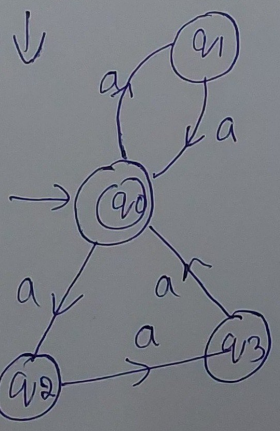
↓

P.T.O



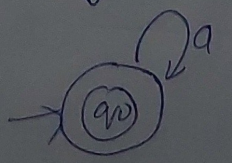
9) $(aa + aaa)^*$



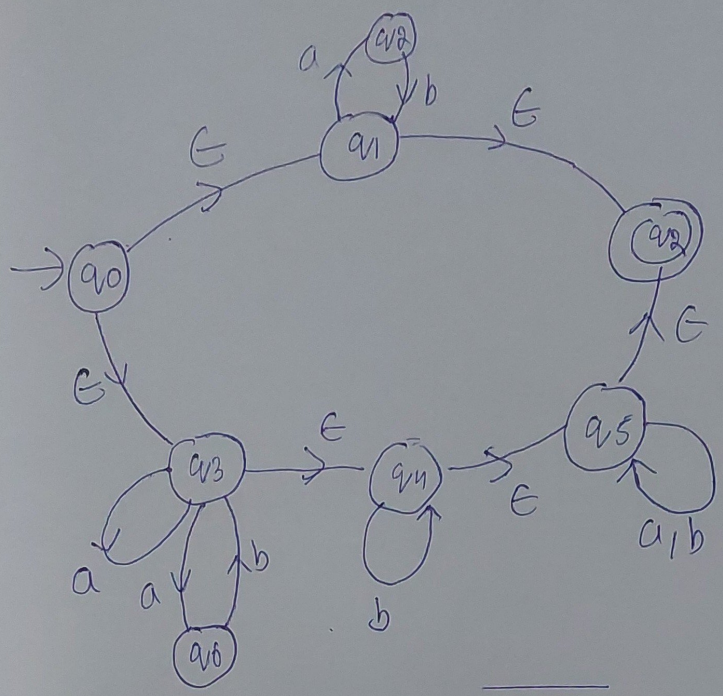


10) $(a + aaaa)^*$

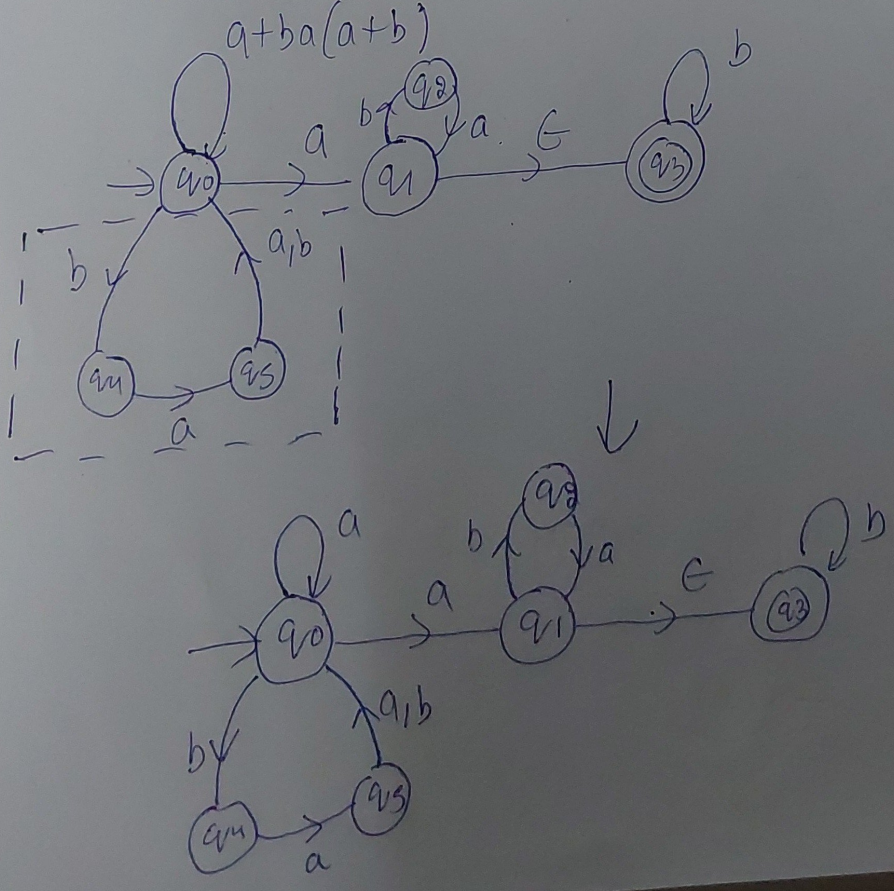
Can be simply written as



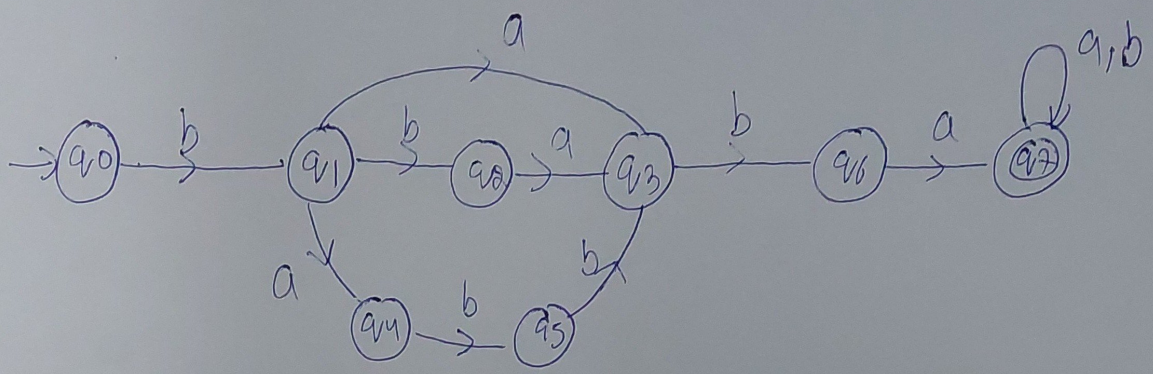
11> $(ab)^* + (a+ab)^* b^* (a+b)^*$



12> $[a + ba(a+b)]^* a(ba)^* b^*$

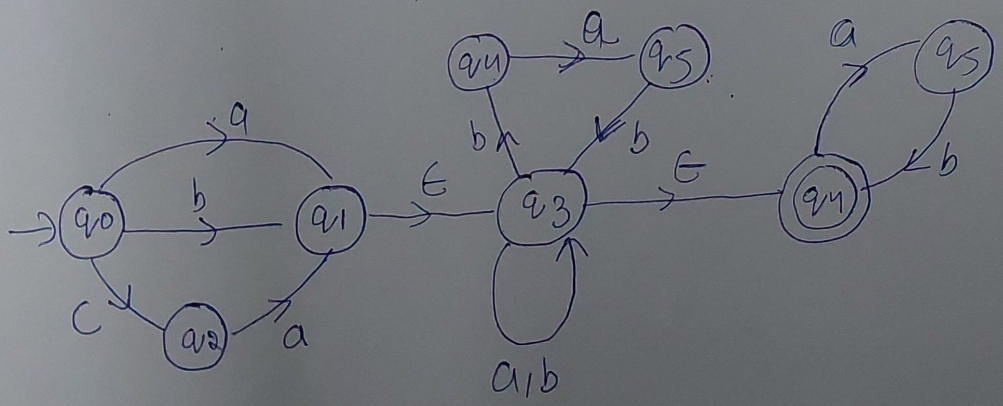


13) $b(a+ba+abb)(ba(a+b)^*)$



---

14) $(a+b+(a)((bab)^*+(a+b)^*)^*(ab)^*$



---o---

↳ "Pumping Lemma for Regular Languages"

↳ It is used to prove that a language is not Regular.
↳ It cannot be used to prove that a language is Regular.

Note :⇒ It is a "negative test"

Pumping Lemma :⇒ If "A" is a Regular Language, then "A" has a pumping length "P" such that any string "S" where $|S| \geq P$ may be divided into 3 parts, $S = xyz$ such that the following conditions must be true:

i) $xy^iz \in A$ for every $i \geq 0$
ii) $|y| > 0$
iii) $|xy| \leq P$

} All these conditions must hold true.

To prove that a language is not Regular using Pumping Lemma, follow the below steps:
(We prove using Contradiction)

↳ Assume that "A" is Regular.
↳ It has to have a Pumping length (say P).
↳ All strings longer than P can be pumped $|S| \geq P$.
↳ Now find a string "S" in A such that $|S| \geq P$.
↳ Divide S in xyz.
↳ Show that $xy^iz \notin A$ for some i
↳ Then consider all the ways that 'S' can be divided into xyz.

↳ Show that none of these can satisfy all the 3 pumping lemma conditions at the same time.

↳ S cannot be pumped == Contradiction

Q1:→ Using Pumping lemma prove that, the language $A = \{a^n b^n \mid n \geq 0\}$ is not Regular.

Sol:→ We already know that $a^n b^n \mid n \geq 0$ is not regular because here we have to compare the number of a's with the number of b's. (we can have any number of a's followed by exactly the same number of b's).

We will prove by Contradiction.

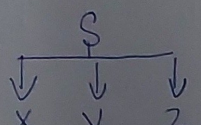
Let's assume that "A" is regular {$a^n b^n \mid n \geq 0$ is regular}

if "A" is a regular language, then it need to have a pumping length. "p" such that any string whose length is greater than or equal to "p" should be pumped.

Let's take a string $S = a^p b^p$ where "p" is pumping length

Now let's divide this S into 3 parts x y & z

Let's now see how to divide this "S" into 3 parts



Pumping

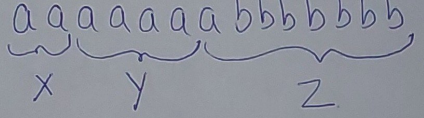
Let $P = 7$ (P is pumping length)

Then

$S =$ "aaaaaaa bbbbbbb" $(P = 7)$

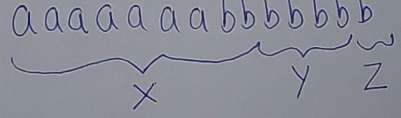
Now let's see all the possibilities in which we can divide this 'S' into 'xyz'

Case ① :⇒ 'y' part is in 'a' part completely.



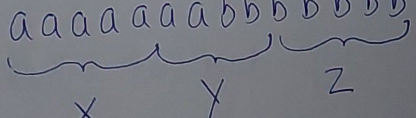
'y' is in 'a' part only

Case ② :⇒ "y" is in "b" part completely



'y' is in 'b' part only

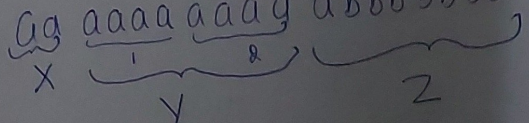
Case ③ :⇒ 'y' is in both 'a' & 'b' part



'y' is in both a & b part

For Case ①, let's take a xy^iz $i = 2$

xy^2z this means we have to repeat our y part two times



xy^2z is not a part of our language.

Since "No. of [a's]" & "no. of [b's]" in the string are not equal, ∴ this string doesn't belong to language. for Case ①

→ Now let's try for Case ②.

let's take xy^iz let $i = 2$

aaaaaaabb bbbb bbbb b

x 1 2 z

y

xy^2z doesn't belong to 'L'

Since "No. of [a's]" & "no. of [b's]" in this string are not equal ∴ this string doesn't belong to language for Case ② also

→ Now let's try for Case ③

let's take xy^iz let $i = 2$

aaaaa aabb aabb bbbbb

x 1 2 z

y

xy^2z does not belong to 'L'

Since "No. of [a's]" & No of [b's]" does not follow the order ∴ the string doesn't belong to lang. for Case ③ also.

↓ It doesn't follow a^nb^n pattern.

Since strings in all the cases after pumping doesn't lie in language, Hence our assumption that "L" is regular was wrong ∴ we can say that 'L' is not regular.

We have to check for other conditions also

for case ① $|xy| \leq p$ is satisfied ($|xy| = 6$)

case ② $|xy| \leq p$ not satisfied ($|xy| = 13$)

case ③ $|xy| \leq p$ not satisfied ($|xy| = 9$)

All the 3 condition have to be satisfied which is not occurring here, hence "L" is not regular.

Q2:→ Using pumping lemma prove that the language $A = \{yy \mid y \in \{0,1\}^*\}$ is not regular.

Sol:→ We will prove it by contradiction.

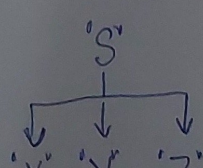
Let's assume that $A = \{yy \mid y \in \{0,1\}^*\}$ is regular.

So, if A is regular than for a pumping length "p" the strings obtained after pumping should belong to the language.

Let's choose a string ['s"] from the language

$s = 0^p 1 0^p 1$ { this string clearly belongs to the lang. A }

Now once we have chosen the string "s", we can divide this string into 3 parts xyz



Let's consider pumping length $p = 7$

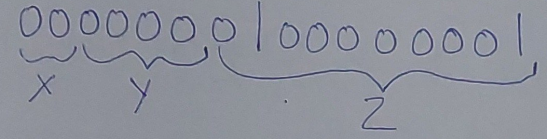
then the string is

0000000|0000001

$0000000|00000001$

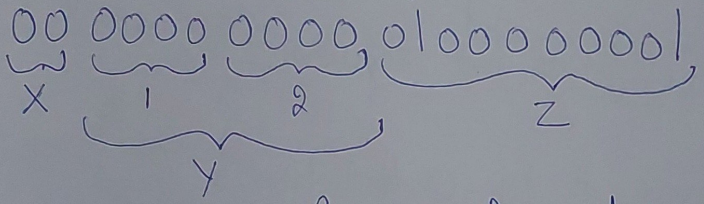
Now let's divide this string into 'X, Y & Z'

let 'y' be at



let's find xy^iz $i=2$

let's find xy^2z



Since xy^2z doesn't belong to A

∴ A is not regular.

& other two conditions, $|y|>0$ & $|xy| \leq p$ are satisfied.

For a lang. to be Regular, all the conditions must be satisfied.

—— o ——

→ "Arden's Theorem" :-> (1)

In order to find out a regular expression of a finite-Automata, we used Arden's theorem along with the properties of regular expressions.

Statement :->

Let 'P" & 'Q' be two regular expressions.

If 'P' does not contain a null string

then $\boxed{"R = Q + RP"}$ has a unique solution

& that solution is $\boxed{R = QP^*}$

Proof of Arden's theorem :->

$R = Q + RP$ Let's put the value of 'R" in R.H.S we get

$\Rightarrow R = Q + (Q+RP)P$

$\Rightarrow R = Q + QP + RPP$ Again putting the value of 'R' in R.H.S we get

$\Rightarrow R = Q + QP + (Q+RP)PP$

$\Rightarrow R = Q + QP + QP^2 + RP^3$.

When we put the value of 'R' in R.H.S recursively, we get the following

P.T.O

$\Rightarrow R = Q + QP + QP^2 + QP^3 + \cdots\cdots$

$\Rightarrow R = Q(\epsilon + P + P^2 + P^3 + \cdots\cdots)$

$\downarrow P^*$

$\therefore \boxed{R = QP^*}$

Hence proved.

Note :⇒ Arden's theorem is used for converting a "finite Automata" into a "regular expression"

Method to apply Arden's theorem :⇒

Assumptions for applying Arden's theorem

1) The transition diagram must not have NULL transitions.

2) It must have only one initial state.

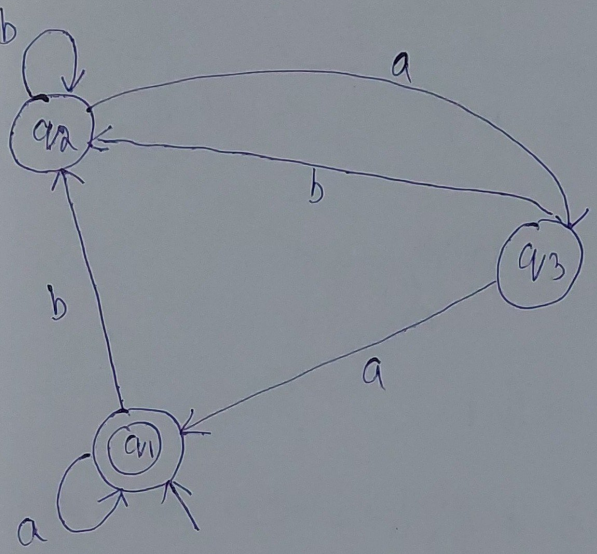
Steps of Arden's theorem :⇒

Step 1 :⇒ Form an equation for each state considering the transitions which comes towards that state.

Step 2 :⇒ Add "ϵ" in the equation of initial state.

Step 3 :⇒ Bring final state in the form of $R = Q + RP$ to get the required regular expression.

Q1:- Construct a regular expression corresponding to the automata given below :- (3)



Solution:- Here the initial state & final state is 'q_1'

The equations for the three states q_1, q_2 & q_3 are as :-

$$q_1 = q_1 a + q_3 a + \epsilon$$

$$q_2 = q_2 b + q_1 b + q_3 b$$

$$q_3 = q_2 a$$

Now, we will solve these 3 equations :-

$$q_2 = q_2 b + q_1 b + q_3 b$$

Substitute the value of q_3 in it

$$q_2 = q_1 b + q_2 b + (q_2 a) b$$

$$q_2 = \underbrace{q_1 b}_{Q} + \underbrace{q_2 (b + ab)}_{R \quad P}$$

$R = Q + RP$, Applying Arden's theorem $R = QP^*$

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

Now let's solve q_1

$$\Rightarrow q_1 = q_1 a + q_3 a + \epsilon$$

Substitute the value of q_3

$$\Rightarrow q_1 = q_1 a + q_2 aa + \epsilon$$

Now substitute the value of q_2

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

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

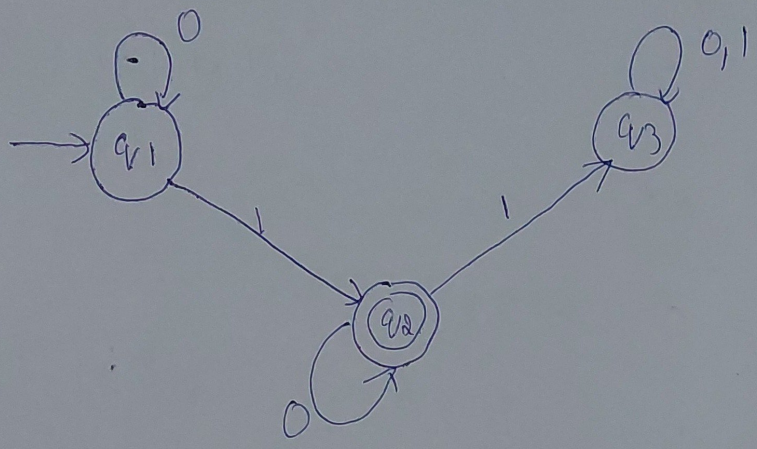
$$\Rightarrow \underbrace{q_1}_{R} = \underbrace{\epsilon}_{Q} + \underbrace{q_1}_{R} \underbrace{(a + b(b+ab)^* aa)}_{P}$$

Apply Arden's theorem

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

$$\Rightarrow q_1 = (a + b(b+ab)^* aa)^* \longrightarrow \text{Regular Expression}$$

Q2:- Construct a regular expression corresponding to the automata given below:- (5)



Sol:- Here the initial state is 'q1' & the final state is q2.

Now let us write the equations for these states:-

$$q_1 = q_1 0 + \epsilon$$

$$q_2 = q_1 1 + q_2 0$$

$$q_3 = q_2 1 + q_3 0 + q_3 1$$

Now let's solve these three equations:-

$$q_1 = q_1 0 + \epsilon$$

$$q_1 = \epsilon + q_1 0$$

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

Apply Arden's theorem

$$\Rightarrow q_1 = \epsilon(0)^* \Rightarrow q_1 = 0^*$$

P.T.O

Let's now solve

(6)-

$$q_2 = q_1 1 + q_2 0$$

Substitute the value of q_1

$$\Rightarrow q_2 = 0^* 1 + q_2 0$$

$$\Rightarrow \underset{R}{q_2} = \underset{Q}{0^* 1} + \underset{R}{q_2} \underset{P}{0}$$

Applying Arden's theorem

$$\Rightarrow \boxed{q_2 = 0^* 1 0^*}$$

∴ "Required Regular Expression"

→ Equivalance of two finite state machines (Automata) :-

Method for comparing two FA's is as explained below:-

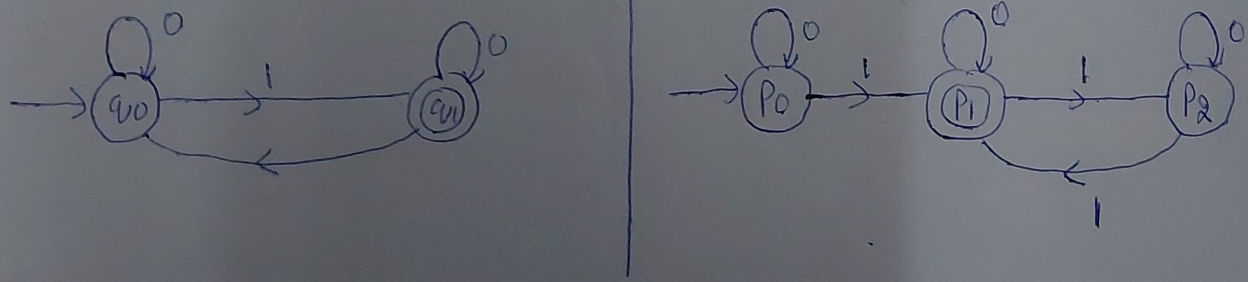
Let "M_1" & "M_2" be two FA's & Σ be a set of input symbols.

Step 1 :- Construct a transition table that has pair with entries (q_1, q_2). where $q_1 \in M_1$ & $q_2 \in M_2$ for each input symbol.

Step 2 :- If we get in a pair as one final state and other non-final state then we terminate the construction of transition table declaring that two FA's are not equivalent.

Step 3 :- The construction of the transition table gets terminated when there is no new pair appearing in the transition table.

Q:- Find out whether the given two DFA's are equivalent or not?



P.T.O

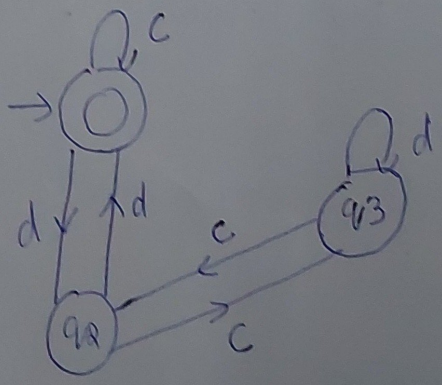
	0	1
$\{q_0 P_0\}$	$\{q_0, P_0\}$	$\{q_1, P_1\}$
$\{q_1, P_1\}$	$\{q_1, P_1\}$	$\{q_0, P_2\}$
$\{q_0, P_2\}$	$\{q_0, P_2\}$	$\{q_1, P_1\}$

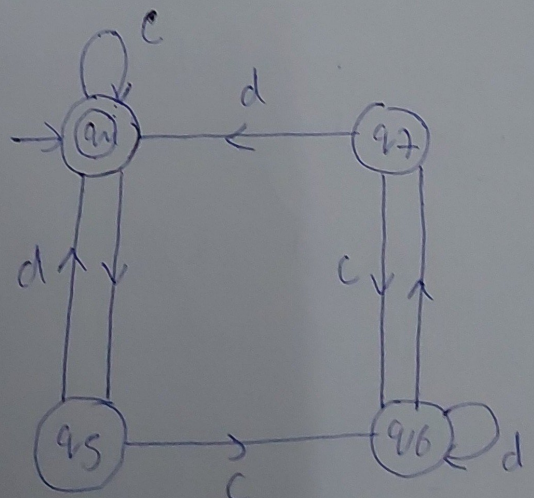
Since no new state has been generated, we will stop here.

Now let's check each pair, if every pair is a combination of only final states or only non-final states, we will say that, the two DFA's are equivalent.

∴ These two DFA's are equivalent.

Q2:→ Find out whether these two DFA's are equivalent or not?





P.T.O

States	"c"	"d"
$\{q_1, q_4\}$	$\{q_1, q_4\}$	$\{q_2, q_5\}$
$\{q_2, q_5\}$	$\{q_3, q_6\}$	$\{q_1, q_4\}$
$\{q_3, q_6\}$	$\{q_2, q_7\}$	$\{q_3, q_6\}$
$\{q_2, q_7\}$	$\{q_3, q_6\}$	$\{q_1, q_4\}$

Since all the pairs comprises of either "Final states" only or "Non-final states" only. we can say that both the DFA's are equivalent.