Unit - 2

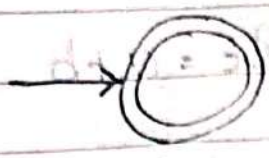
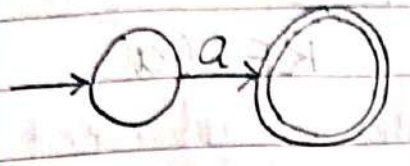
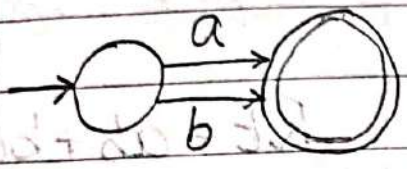
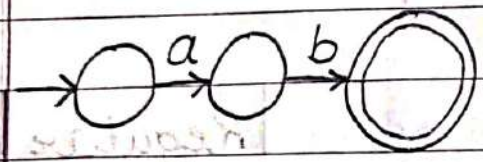
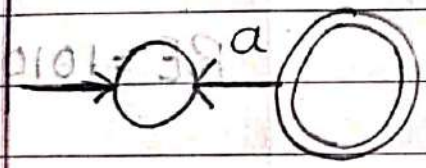
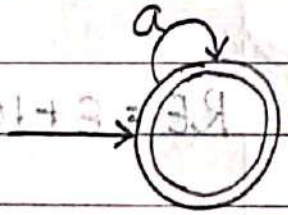
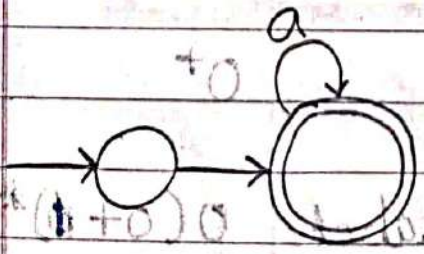
Regular Expressions and Languages

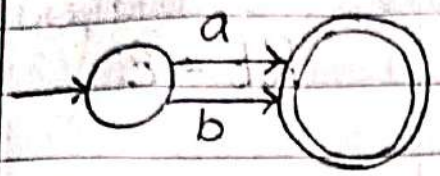
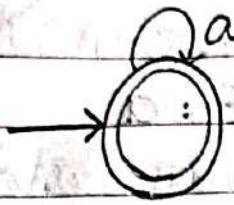
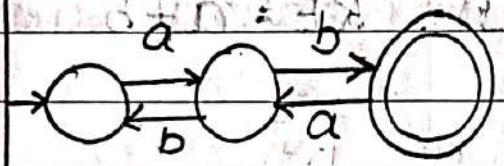
☆ Regular Expressions Introduction

- Regular Language : The set of strings accepted by finite automata is known as regular language.
- Such a language can also be represented in compact form using set of operators.
- These operators are :-

1 + Union
2 . Concatenation
3 * Star or Closure

- Regular Expression is written using set of operators such as [+.*] & regular language is known as a regular language.
- They are used to represent the regular language in compact form.

Automata	Language	Regular Expression
	$\{\varepsilon\}$	$RE = \varepsilon$
	$\{a\}$	$RE = a$
	$\{a, b\}$	$RE = a + b$
	$\{a, b\}$	$RE = a \cdot b$
	ϕ	$RE = \phi$
	$\{\varepsilon, a, aa, aaa \ldots\}$	$RE = a^*$
	$\{a, aa, aaa, \ldots\}$	$RE = aa^*$ or a^*

Automata	Language	Regular Expression
	{a, b}	RE = a+b
	{ε, a, b, aa, ab, ba aaa, ...}	RE = (a
	{ab, ba}	RE = ab + ba

Regular Language	Regular Expression
1] The set {1010}	RE = 1010
2] The set {10, 1010}	RE = 10 + 1010
3] The set {ε, 10, 01}	RE = ε + 10 + 01
4] The set {ε, 0, 00, 000....}	0^*
5] The set {0, 00, 000....}	0^+
6] The set of strings over alphabet {0, 1} starting with 0	$0(0+1)^*$

7	The set of strings over alphabet {a,b} starting with a and ending with b.	a(a+b)*b
8	The set of strings over alphabet {0,1} ending with 1	(a+b)*b.
9	The set of string Recognized by $(a+b)^3$	(a+b)(a+b)(a+b)

Precedence of Operators

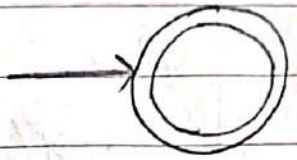
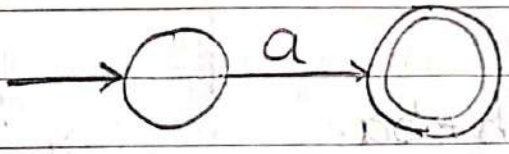
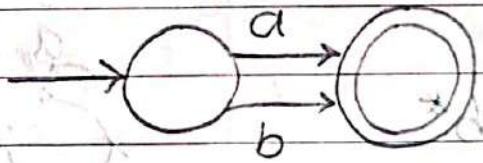
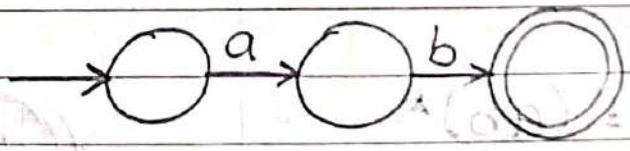
- Similar to arithmetic operators of regular expression follows a predefined precedence
- Star (*) it has the highest precedence in a regular expression
- Dot (.) has the next highest precedence in a regular expression
- Union (+) has the lowest precedence in a regular expression

FA Representing RE

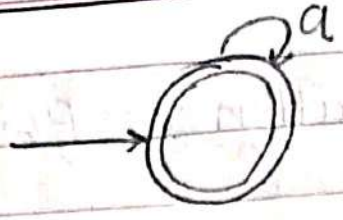
- Finite automata and regular expression are equivalent
- We can express a DFA as an equivalent RE
- We can express a RE as an ε - NFA using operator an alphabet, ε, ϕ.
- The ε - NFA can be converted to DFA and the DFA to an NFA.
- The RE will be equivalent to all the automata we have described in Example.
- Recursive nature of RE
- $R = [1+0[10]^*]^*$ so it is replace in RE
- $1+0[10]^*$ by P so $R = P^*$
- P as $P1 + P2$ where $P1 = 1$ and $P2 = 0[10]^*$
- P2 can be written as $P3 . P4$ where $P3 = 0$ and $P4 = [10]^*$
- P4 can be written as $P5^*$ where $P5 = 10$

- P_5 can be written as $P_6 . P_7$ where P_6 is 1 and P_7 is 0.

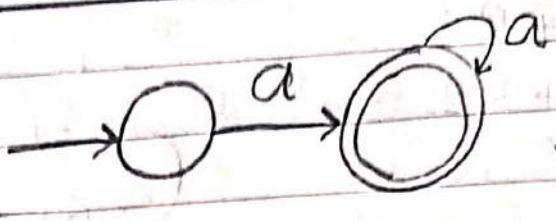
- 1] R1+R2
  2] R1+R2
  3] R1*
  Then we can construct Fas for any RE.

Regular Expression	Automata
1] RE = ε	
2] RE = a	
3] RE = a+b	
4] RE = a.b	
5] RE = ϕ	

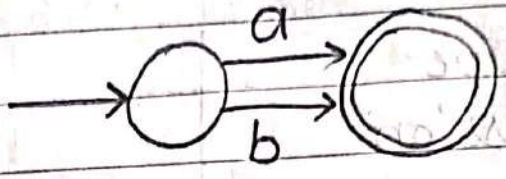
6. RE = a^*



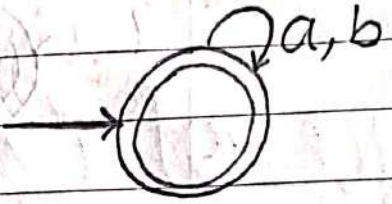
7 RE = aa^* or a^+



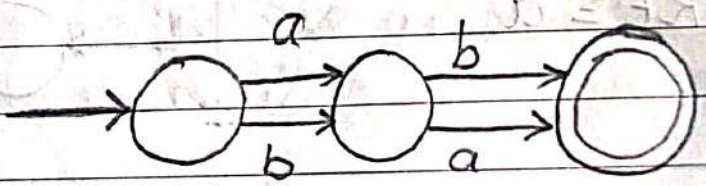
8] RE = a+b



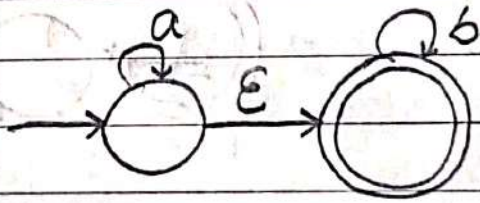
9] RE = (a+b)*



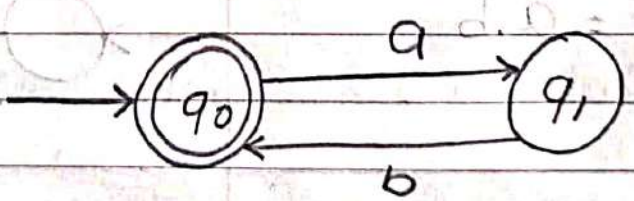
10] RE = ab+ba



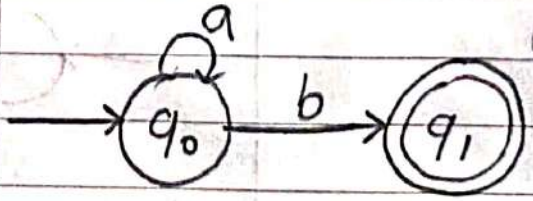
11] RE = a^*b^*



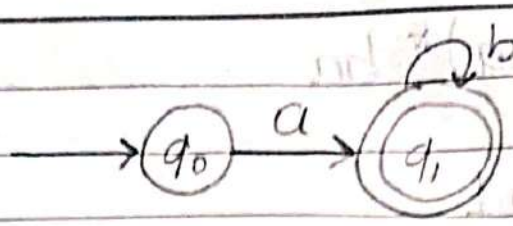
12] RE = (ab)*



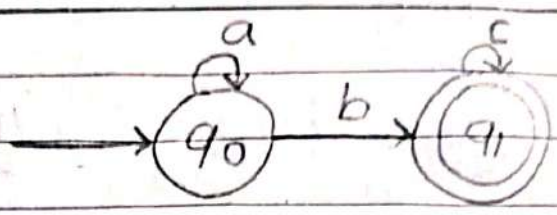
13] RE = a^*b



14] RE = ab^*



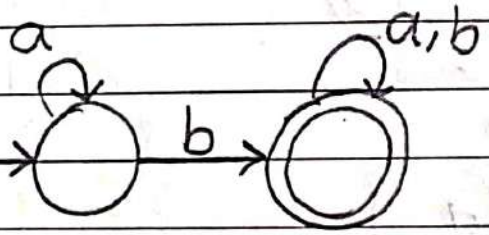
15] RE = $a^* bc^*$



Examples on RE to FAs Conversion

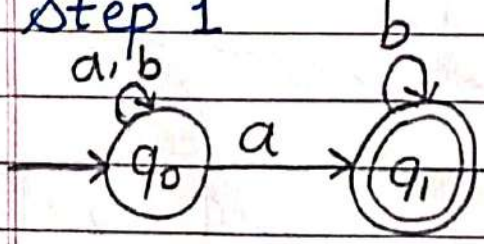
① Convert the following RE to FAS

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

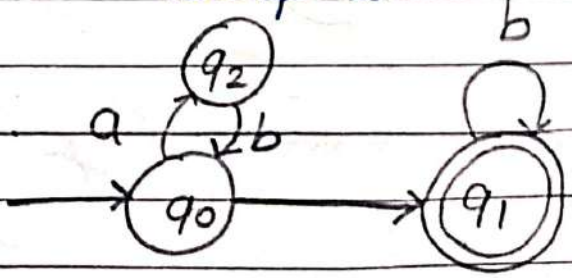


② $(ab)^* ab^*$

Step 1

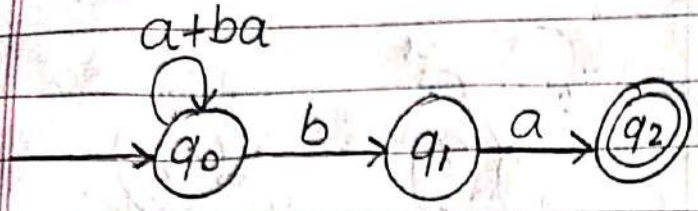


Step 2

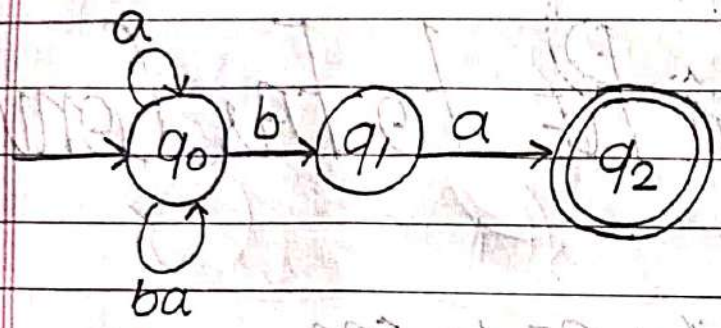


③ $(a+ba)^* ba$

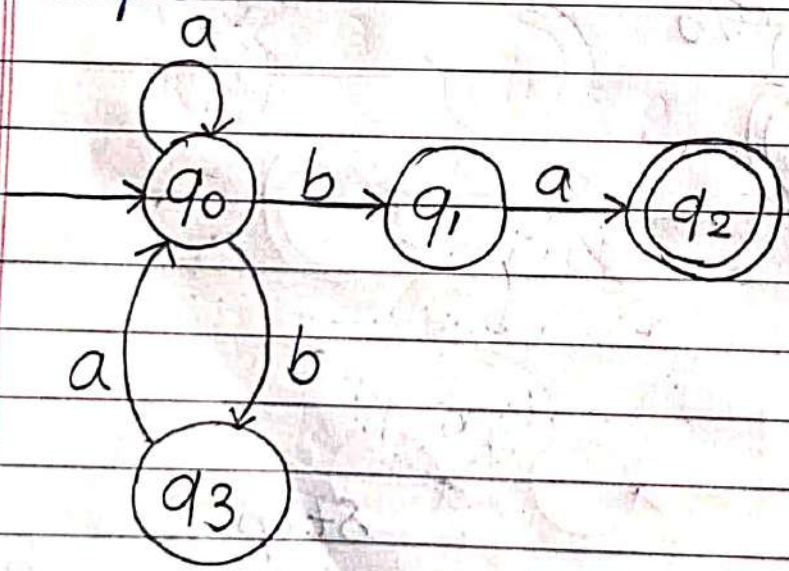
Step 1



Step 2

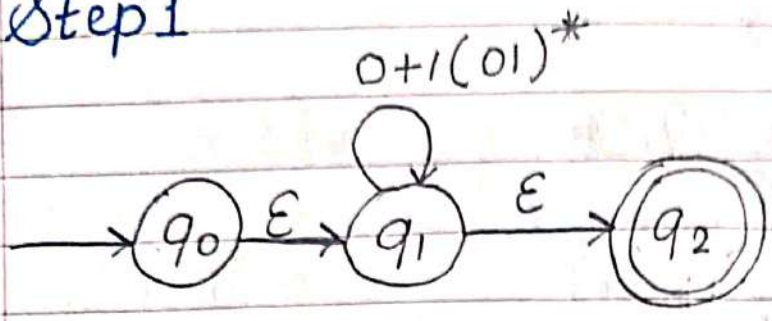


Step 3

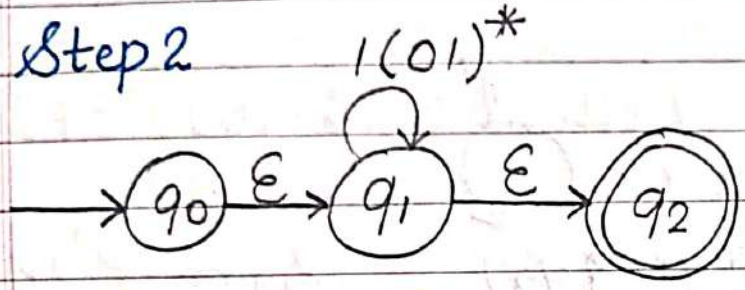


④ $(0+1[(01)^*)^*$

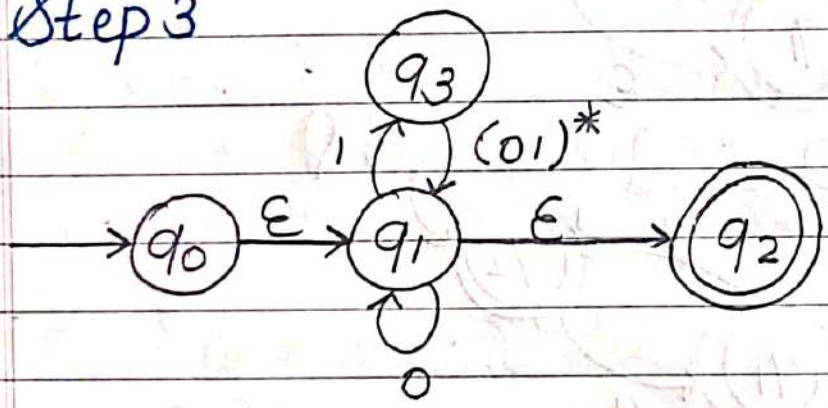
Step 1



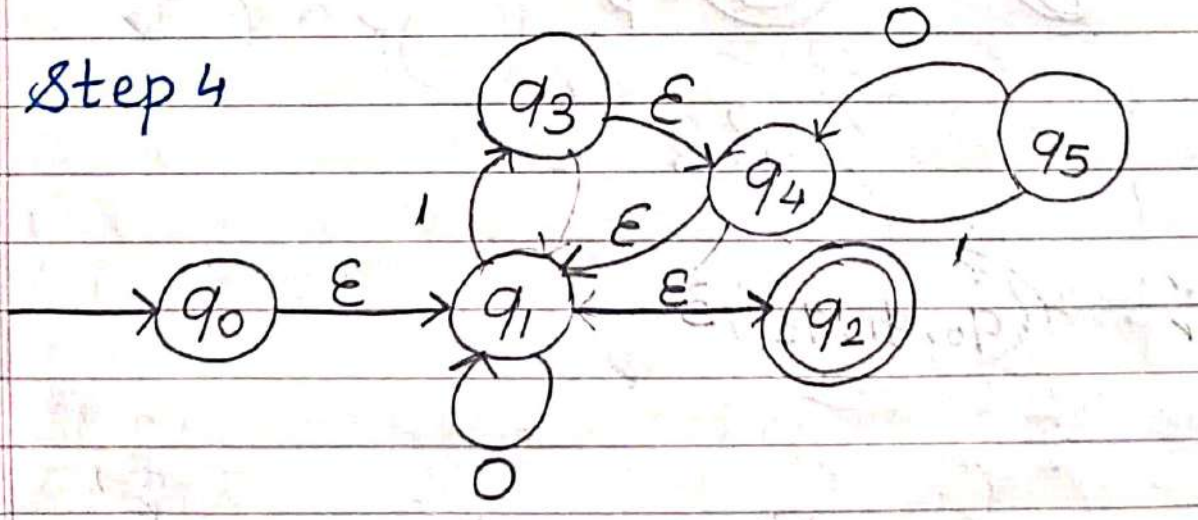
Step 2



Step 3

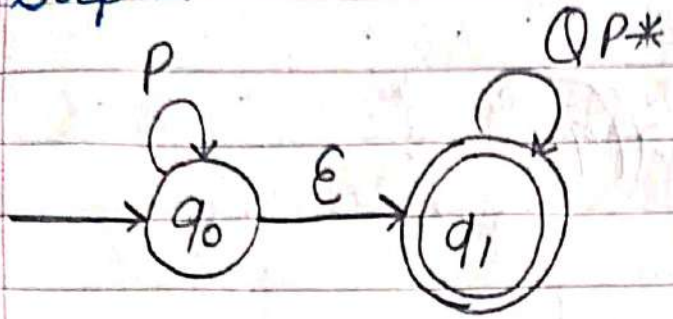


Step 4

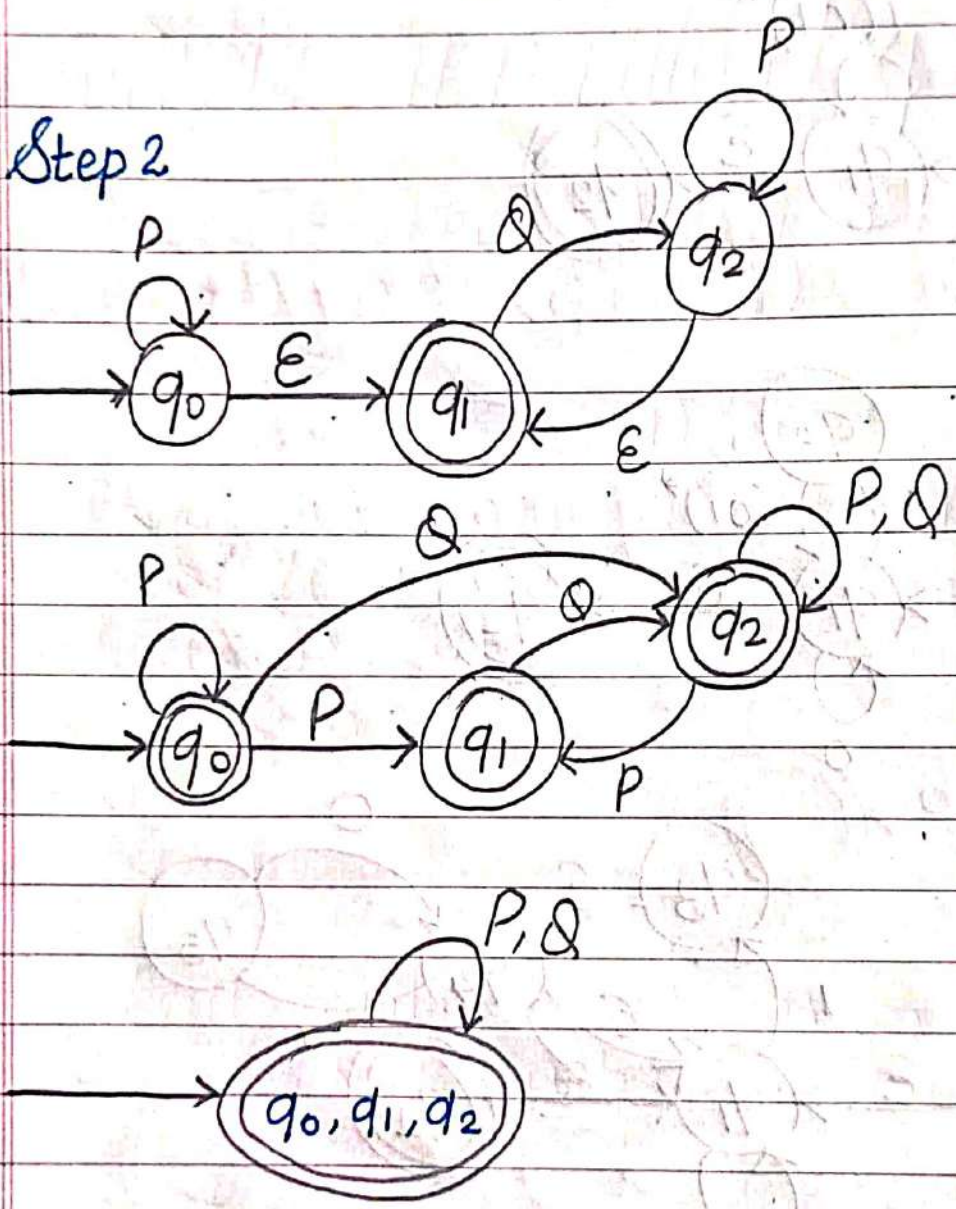


(5) Show that $p^*(QP)^* = (P+Q)^*$

Step 1



Step 2



So $p^*(QP^*)^* = (P+Q)^*$

Determination of RE

- A regular expression over alphabet $q = (a1, a2, a3, \ldots an)$ is defined recursively as follows

1] A null string epsilon is a RE
2] An Empty set is a RE
3] An alphabet is a RE
4] If X and Y are RE then their concatenation is a RE
5] Closure of $X(X^*)$ is a RE

Basic Property of RE

If P, Q & R are RE then basic properties of RE are as given :-

1] $\phi + R = R$
2] $\phi \cdot R = R \cdot \phi = R$
3] $\epsilon \cdot R = R \cdot \epsilon = R$
4] $\epsilon^* = \epsilon$
5] $\phi^* = \epsilon$
6] $R + R = R$
7] $PQ + PR = P(Q+R)$
8] $QP + RP = (Q+R)P$
9] $R^* R^* = R^*$
10] $RR^* = R^* R$

11] $(R^*)^* = R^*$
12] $\varepsilon + RR^* = R^*$
13] $(PQ)^* P = P(QP)^*$
14] $(P+Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$
15] $(P^* Q)^* = \varepsilon + (P+Q)^* Q$

Example on Regular Expression

① Prove that $(1+00^*1)+(1+00^*1)(0+10^*1)+(0+10^*1) = 0^*1(0+10^*1)$

$LHS = (1+00^*1)+(1+00^*1)(0+10^*1)^*(0+10^*1)$
$= (1+00^*1)(\varepsilon+(0+10^*1)^*(0+10^*1))$
$= (1+00^*1)(0+10^*1)^*$
$= [(\varepsilon+00)^*](0+10^*1)^*$
$= 0^*1[0+10^*1)^*$
$= RHS$

② Find all regular expressions for following languages over $\{a,b\}$

1] The set of all string ending in b.
→ $(a+b)^*b$

2] The set of all string ending in ba
→ $(a+b)^*ba$

3] The set of all string ending neither in b nor in ba

→ $(a+b)^* aa + bb$

4] The set of all strings ending in ab.

→ $(a+b)^* ab$

5] The set of all strings ending neither in ab nor in ba.

→ $\epsilon + a + b + (a+b)^* (aa+bb)$

Q3] Find all regular expressions for following subset of $\{0,1\}^*$

① The language of all strings containing exactly two 0's.

→ RE → $1^* 0 1^* 0 1^*$

② The language of all strings containing at least two 0's.

→ RE → $1^* 0 1^* 0 (1+0)^*$

③ The language of all strings that do not end with 01.

→ RE → $(1+0)^* (00 + 11 + 10)$

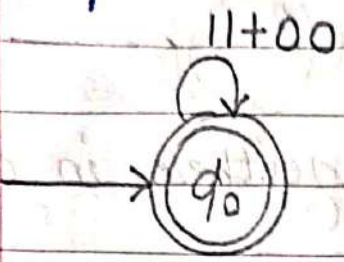
④ The language of all strings starting with 11

→ RE → $11(0+1)^*$

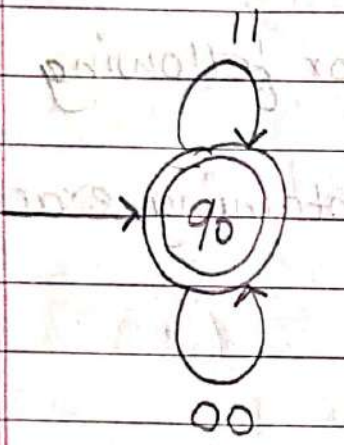
∘ Draw RE to DFA

① $(11+00)^*$

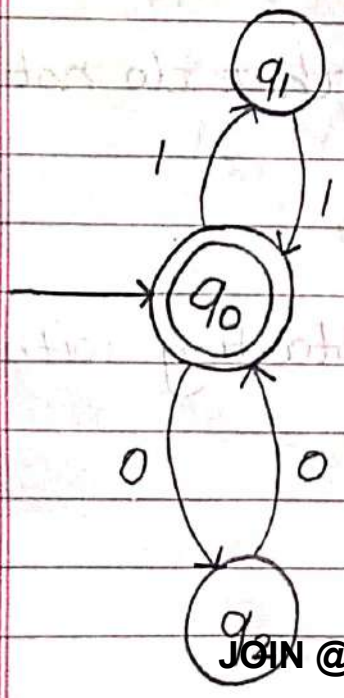
★ Step 1



★ Step 2

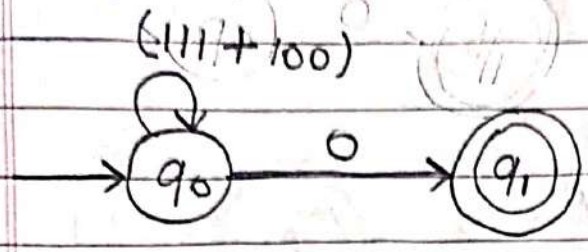


★ Step 3

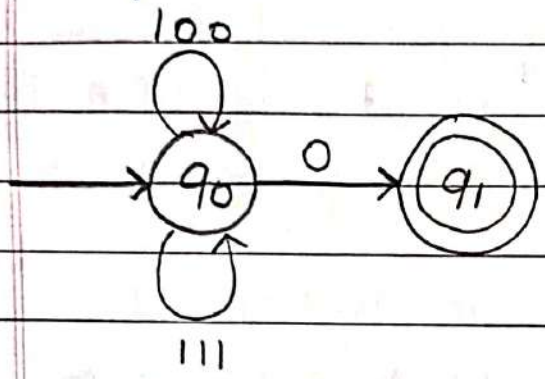


2] $(111+100)^*0$

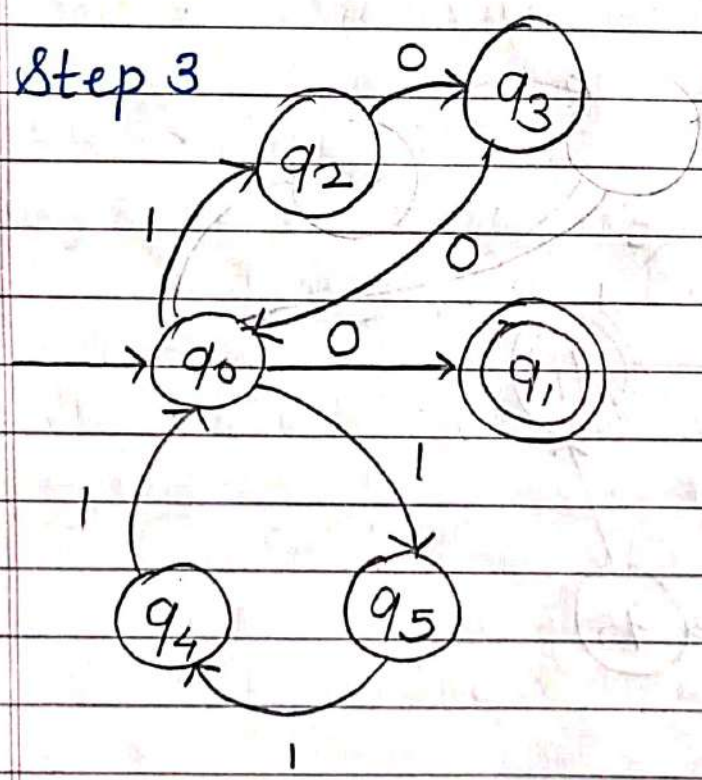
★ Step 1



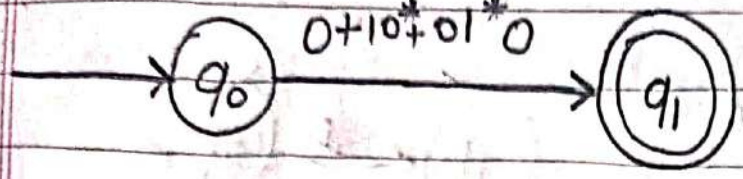
★ Step 2

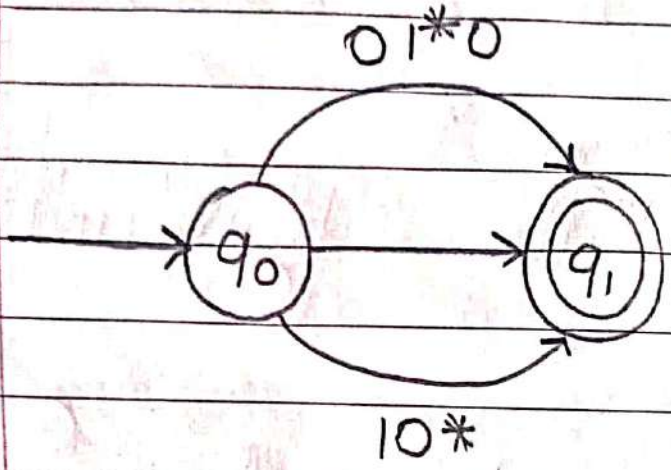


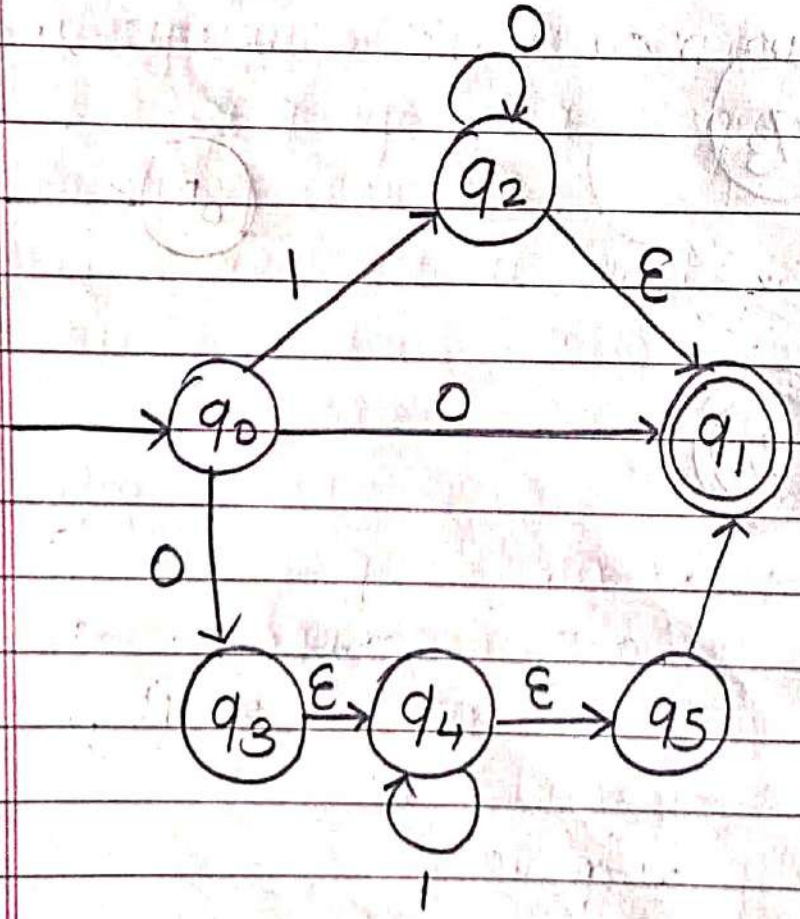
★ Step 3



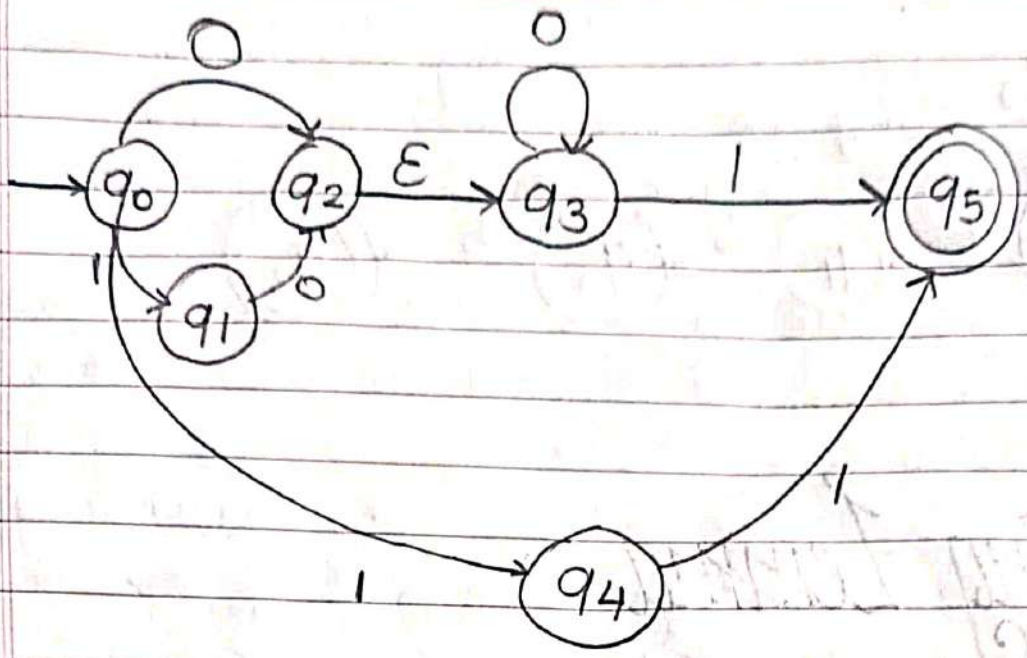
3] $0+10^*+01^*0$



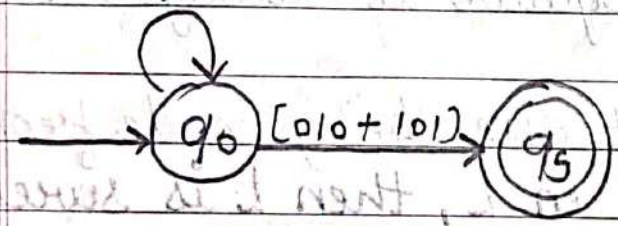


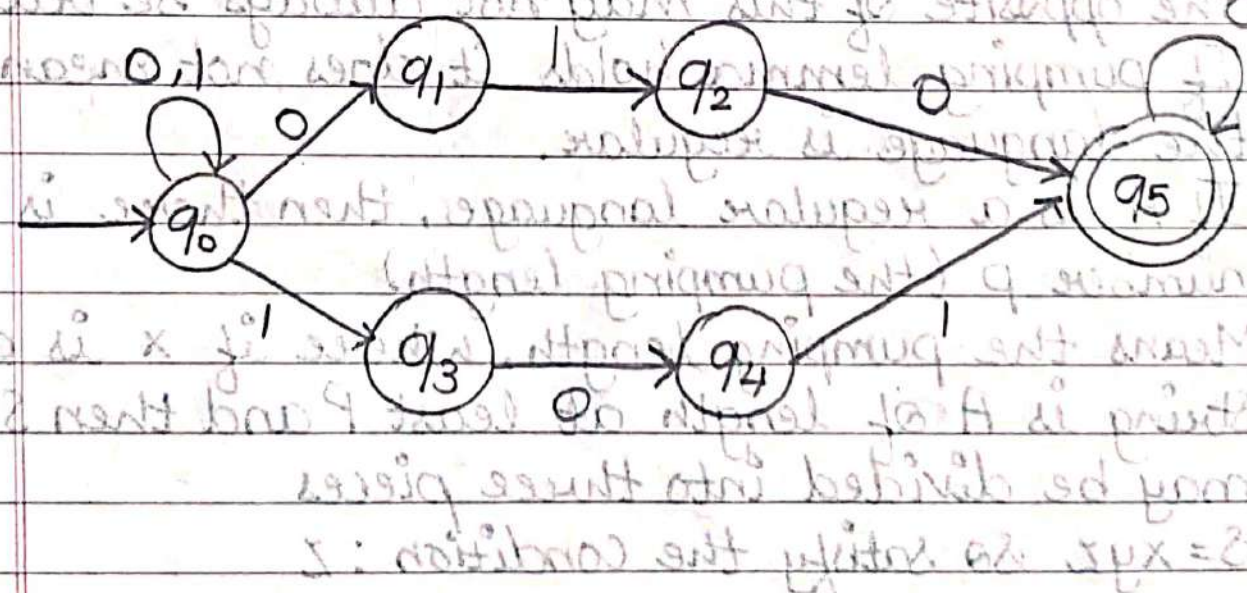


Q4] $10+(0+11)0^*1$



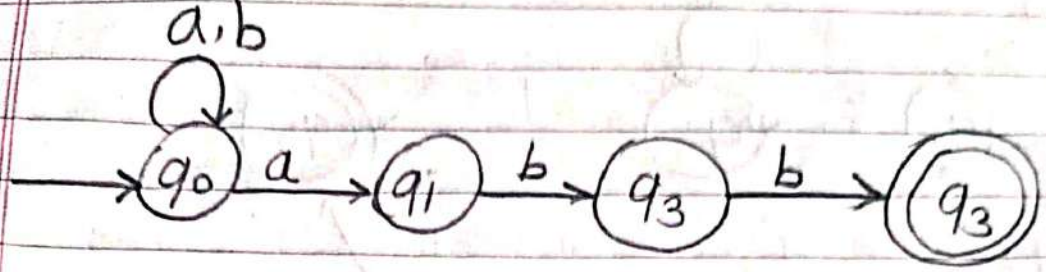
Q5] Find FA for given RE $(0+1)^*.(010+101).(0+1)^*$





Q5] Find FA for given RE

i) $(a+b)^* abb$



Pumping Lemma

- Pumping lemma is used as a proof for irregularity of a language.
- If a language is a regular it always satisfy pumping lemma
- If there exist at least one string made from pumping which is not in L, then L is surely not regular.
- The opposite of this may not always be true. If pumping lemma holds it does not mean that the language is regular.
- If A is a regular language, then there is a number p (the pumping length)
- Means the pumping length, where if x is any string is A of length at least P and then S may be divided into three pieces
  $S = xyz$ so satisfy the condition : z

① For each $i \geq 0$, $xy^i z \in A$
② $|y| > 0$, and
③ $|xy| \leq P$,

The pumping lemma says that is a language A is Regular then any string in the language will have a certain property. Provided that it is long enough, that is longer than length

- Condition 1: For each $i \geq 0$, $xy^i z \in A$ $xyz \in Z$
- Condition 2: $|y| > 0$
- Condition 3: $|xy| \leq P$

☆ Steps to find the language is Regular or Not

① Assume that A is regular language.
② It has to have a pumping length (say P)
③ All string longer than P can be pump $|s| \geq P$
④ Now find string s in a such that $|s| \geq P$
⑤ Divide s into P parts x y z, show that $xyz \notin A$ for some i
⑥ Consider always that s can be divided into x, y, z
⑦ Show that none of these can be satisfy all three pumping condition at the same time
⑧ s can not be pump so it is a contradiction

Example 1

Q Using pumping lemma show that the languag $L = \{ a^n b^{2n} \}$ is not reegulare.

Solution

1] Let us assume that L is a reegulare language And L is accepted by FA with n states.

2] Let us choose a streing reepreesented by given language L & accepted by FA:. Acioeding to language no of occurences of b. So let us consider the streing.

$L = a^n b^{2n}$ $|w| = aabbbb$

Now let us write w as xyz with $|y| > 0$ and $|xy| <= n$

$\underbrace{aa}_{x}\ \underbrace{bb}_{y}\ \underbrace{bb}_{z}$

Now let us pump they yi times write $i > 0$ let us assume $i = 2$

$\underbrace{aa}_{x}\ \underbrace{bbbb}_{y^i}\ \underbrace{bb}_{z}$

Now let us check xy^2z belongs to L are Not

$xy^2z = aa\,bbbbbb$

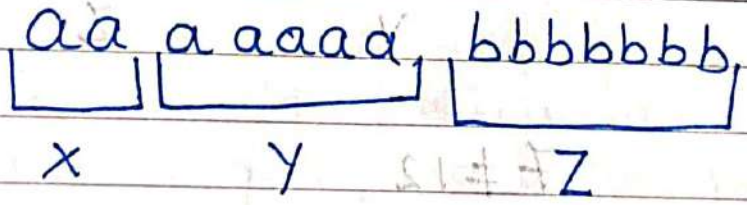
So as per given language no of occurences of a should be followed by double no of occurences of b is not followed by string after pumping so which indicates $|w|$ is not regular language.

Example 2

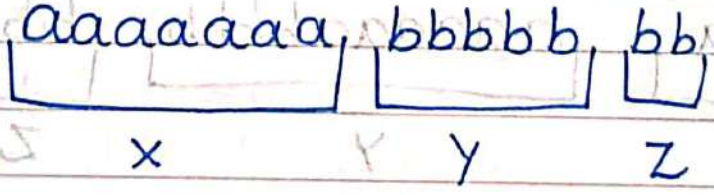
Using pumping lemma show that the language $L = \{a^n b^n\}$ is not regular

$|s| = a^7 b^7 = aaaaaaabbbbbbb$

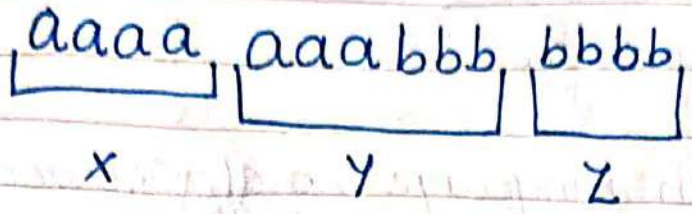
Case 1



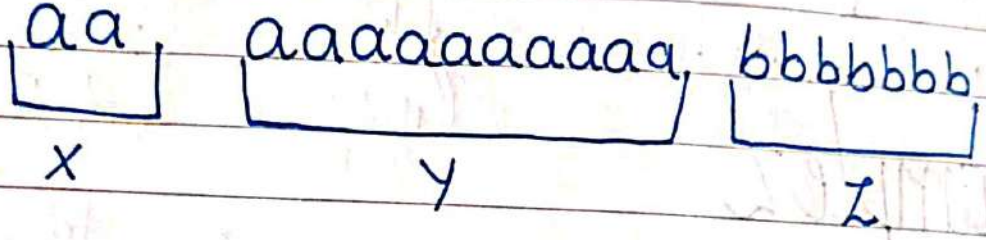
Case 2



Case 3

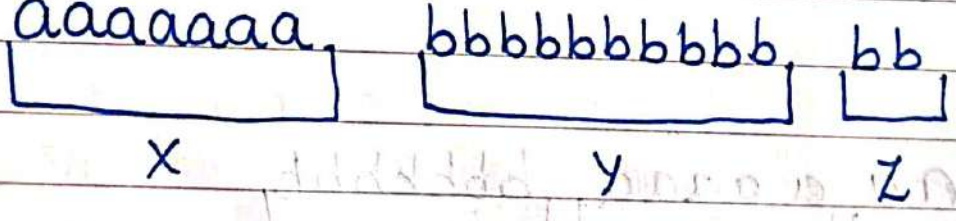


① Case 1



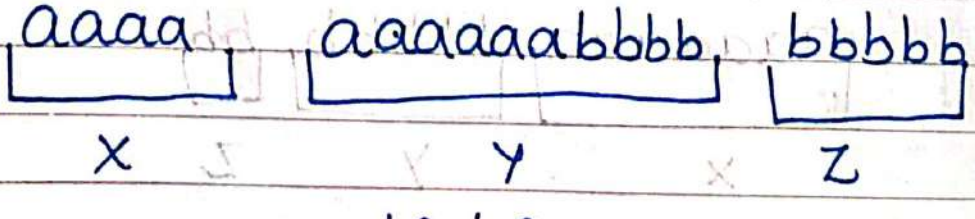
$12 \neq 7$

② Case 2



$7 \neq 12$

③ Case 3



$10 \neq 9$

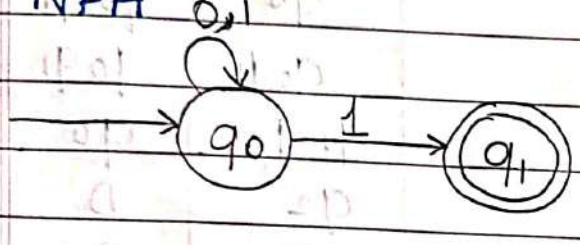
Example

★ Conversion NFA to DFA

L = { set of all string over [0,1] that ends with 1 }

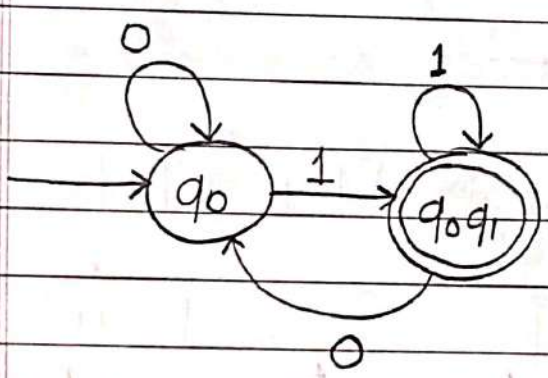
Solution

★ NFA



	0	1
q_0	q_0	q_0, q_1
q_1	ϕ	ϕ

★ DFA

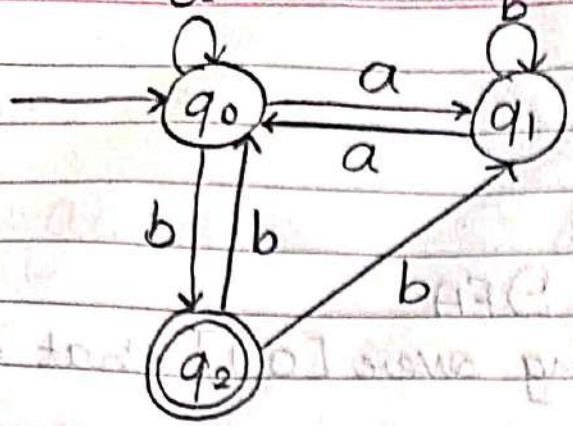


	0	1
q_0	q_0	q_0q_1
q_0q_1	q_0	q_0q_1

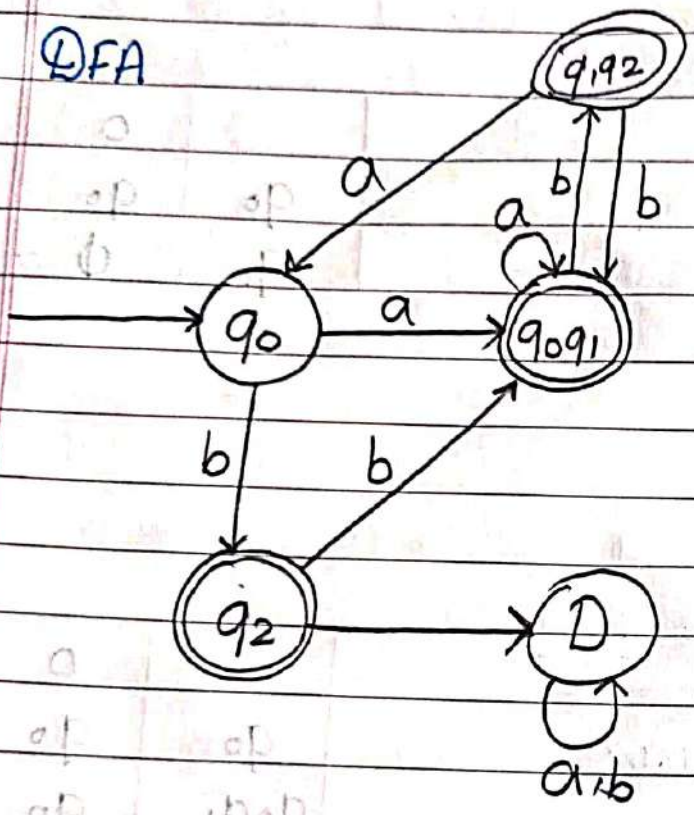
★ Find the equivalent DFA for NFA given by

{M = {q_0, q_1, q_2}, (a, b) δ, q_0, {q_2}}

	a	b
q_0	q_0, q_1	q_2
q_1	q_0	q_1
q_2	ϕ	q_0, q_1

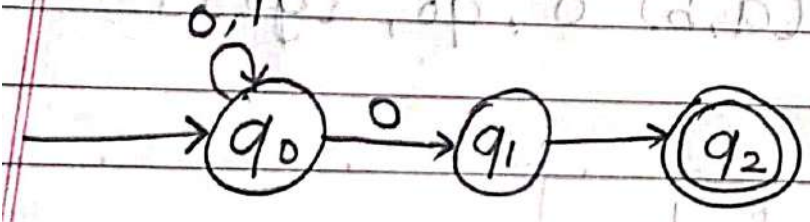


DFA

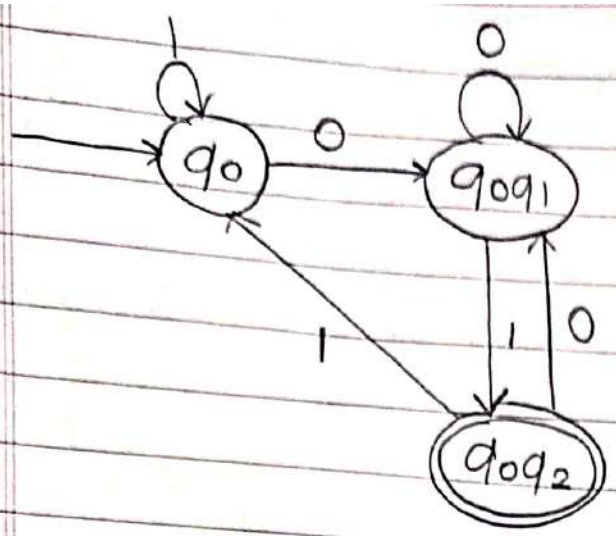


	a	b
q_0	q_0q_1	q_2
q_0q_1	q_0q_1	q_1q_2
q_1q_2	q_0	q_0q_1
q_2	D	q_0q_1
D	D	D

Given below is the NFA for a language L = {set of all the string over (0,1) that ends with 01} construct its equivalent DFA

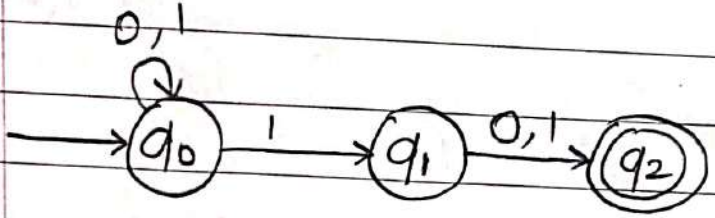


	0	1
q_0	q_0, q_1	q_0
q_1	ϕ	q_2
q_2	ϕ	ϕ

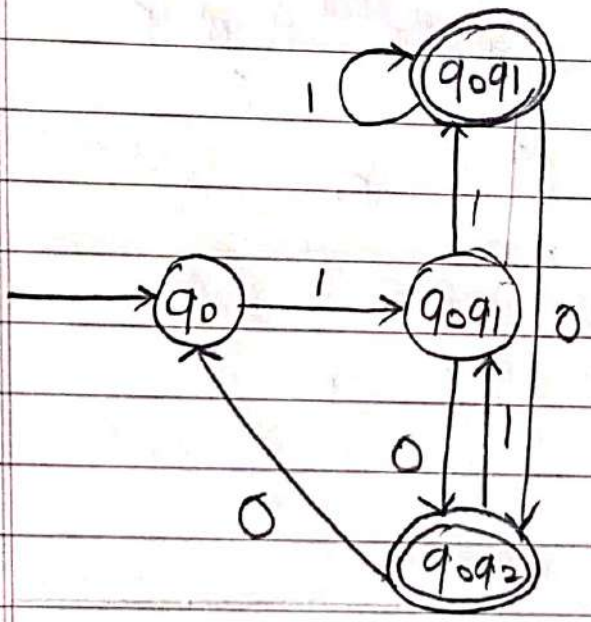


	0	1
q_0	q_0q_1	q_0
q_0q_1	q_0q_1	q_0q_2
q_0q_2	q_0q_1	q_0

★ Design a NFA for a language that accept all string over [0,1] in which the second last symbol is always 1 then convert it to its equivalent DFA.



	0	1
q_0	q_0	q_0, q_1
q_1	q_2	q_2
q_2	ϕ	ϕ



	0	1
q_0	q_0	q_0q_1
q_0q_1	q_0q_2	$q_0q_1q_2$
q_0q_2	q_0	q_0q_1
$q_0q_1q_2$	q_0q_2	$q_0q_1q_2$