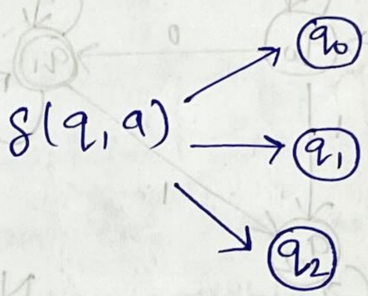
NON-DETERMINISTIC FINITE AUTOMATA (NFA)

Non Determinism means a choice of moves for an automaton. Rather than prescribing a unique move in each situation, we allow a set of possible moves i.e for DFA $\delta(q,a)$ is a single state whereas for NFA $\delta(q,a)$ is (perhaps empty) set of states. The concept of non-determinism play a central role in both the theory of languages and computation.

$\delta(q,a) \longrightarrow q_2$

For DFA



For NFA

The idea of introducing NFA is to simplify the method of designing a DFA. There are several problems and situations when we do not know what would be the next move or exactly next state. In such a case, several moves are possible. We start with one end and follow it until the choice becomes the right choice. If choice does not come out to be right, we retreat to the last decision and explore the other choices. Thus, NFA may be regarded as a parallel computer where several processes can run simultan

A formal definition of NFA is as foll

NFA is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$

where Q = finite set of states

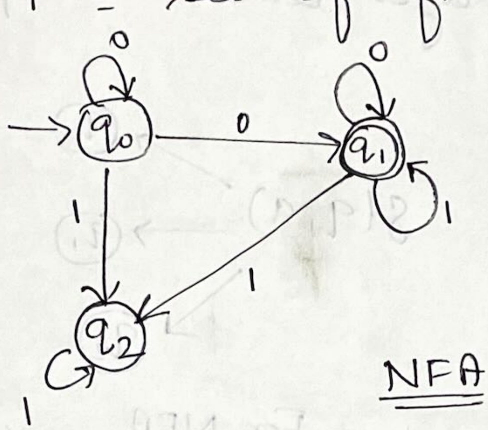
Σ = Input set of alphabet

$\delta = Q \times \Sigma \rightarrow 2^{Q}$

q_0 = Initial state

F = set of final states

e.g:-



NFA

The NFA is basically used in the theory of computation because they are more flexible and easier to use than the DFA's

	0	1
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_1	q_1, q_2
$* q_2$	—	q_2

Difference between NFA and DFA

1) In DFA for a given State, on a given input we reach to deterministic and unique state. On the other hand, in NFA we may lead to more than one States for given i/p.

2) The DFA is a subset of NFA. We need to Convert NFA to DFA in the design of compiler.

3) In non-deterministic automata, the range of δ is the power set of 2^Q, so that its value is not a single element of Q but a Subset of it. This subset defines the set of all possible States that can be reached by transition. But in DFA, it is not possible.

4) We can allow in NDFA or NFA $\delta(q_0, \wedge) = Q$. This means that NFA can make a transition without consuming an input symbol. In DFA, the system changes State if

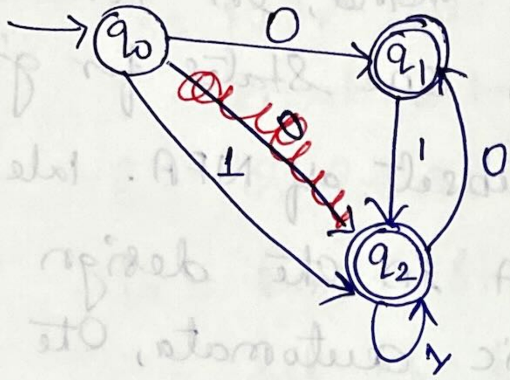
$$\delta(q_i, a) = q_j$$

DFA does not change state without consuming any letter.

5) In an NFA, the set $\delta(q_i, a)$ may be empty i.e there is no transition defined for this situation.

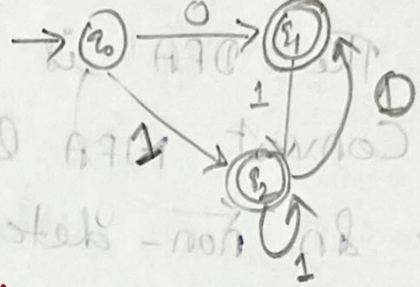
Q: Construct ~~a~~ NFA accepting all strings over $\{0,1\}$ which end in 1 but does
Contain the substring 00.

Sol^n:



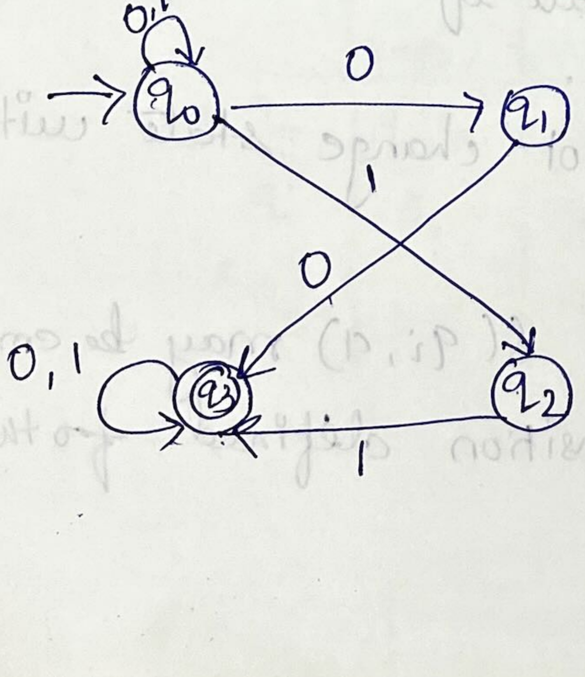
01110101

eg: 01, 0, 1, 10, 11

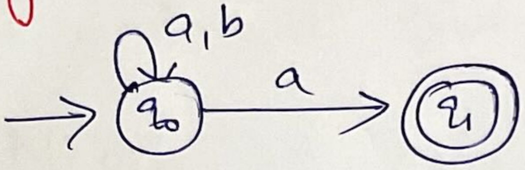


Q: Sketch the NFA state diagram for
$M = \{\{q_0, q_1, q_2, q_3\}, \{0,1\}, \delta, q_0, \{q_3\}\}$ with
the state table as,

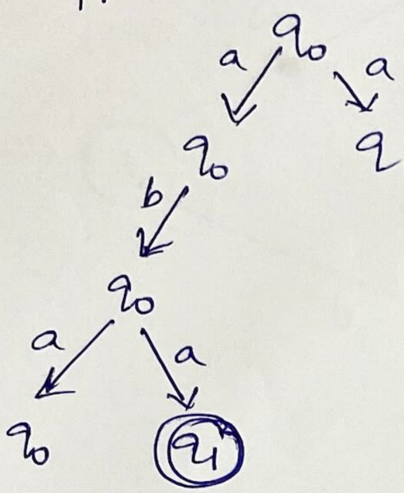
State	Input	
	0	1
$\rightarrow q_0$	$q_0 q_1$	$q_0 q_2$
q_1	q_3	
q_2		q_3
$* q_3$	q_3	q_3



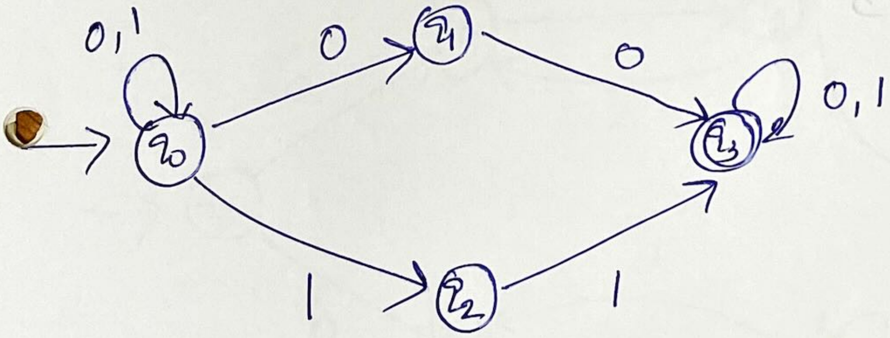
Q: Design a NFA that ends with a over i/p Σ(a,b).



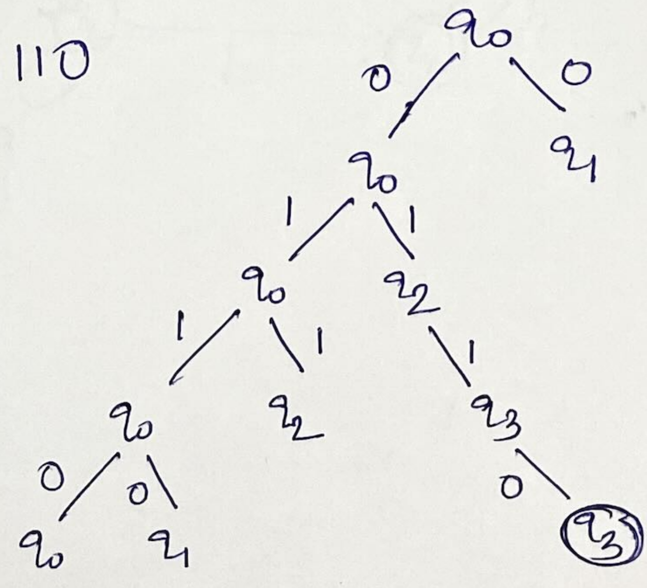
Suppose the string is aba.



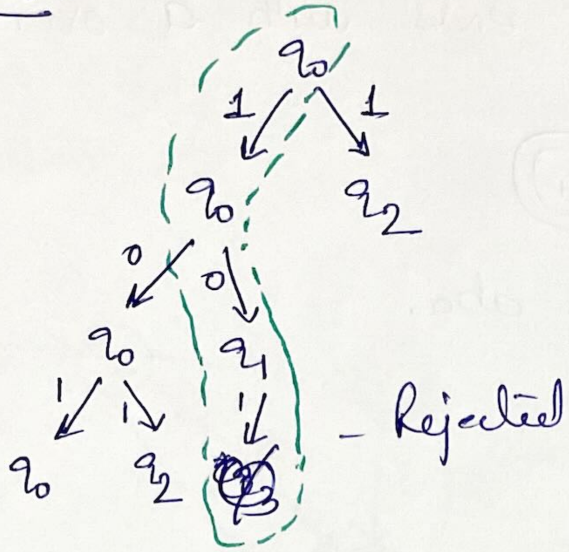
Q: Design NFA for binary strings over Σ(0,1) that contains 00 or 11 as substring.



String: 0110

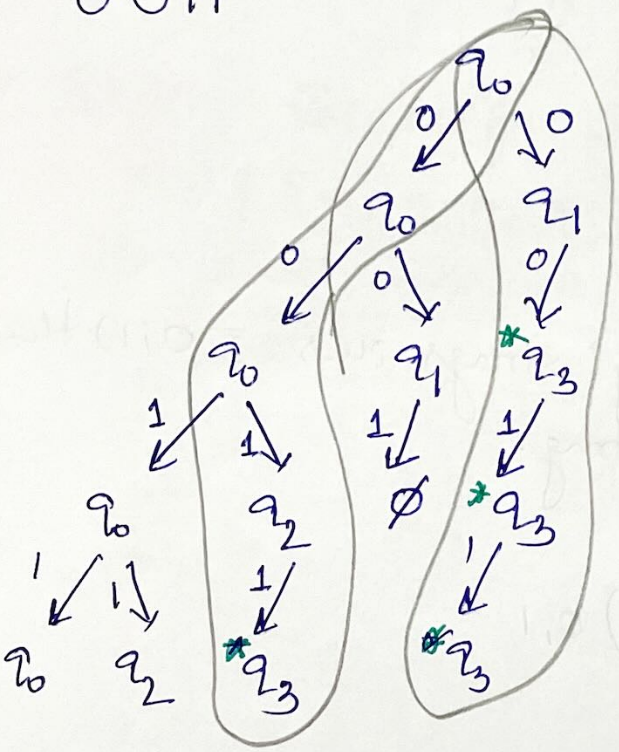


1011



q_0
1
1
q_0
q_2
0
0
q_0
q_1
1
1
1
q_0
q_2
q_3
- Rejected

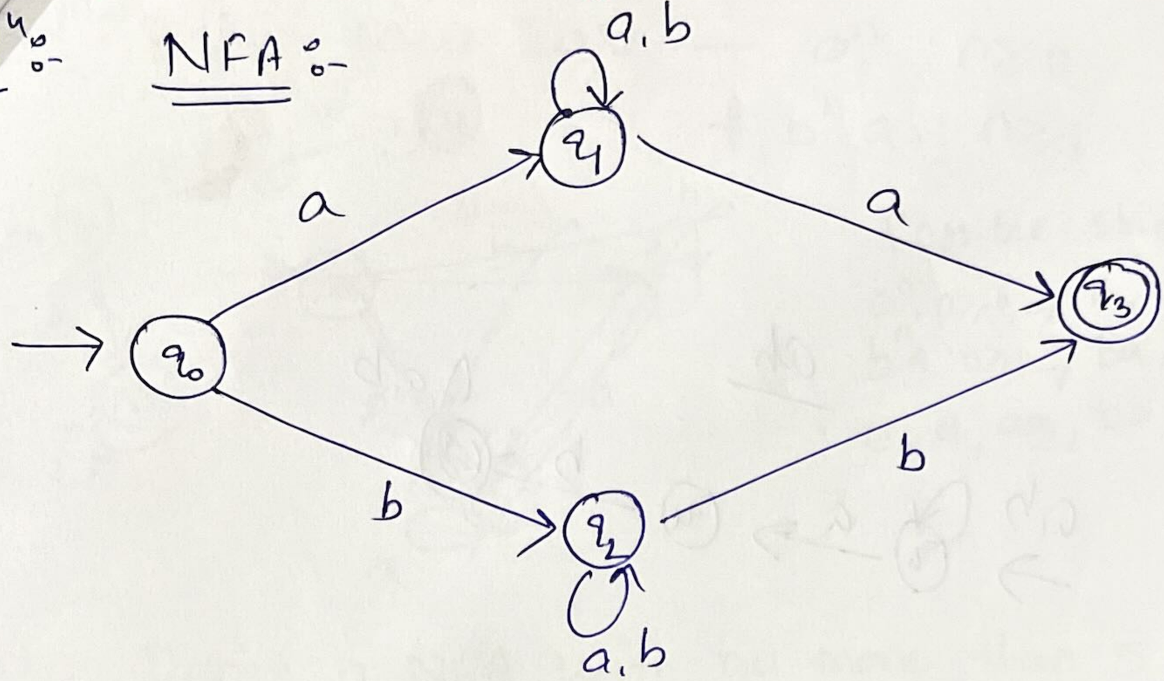
0011



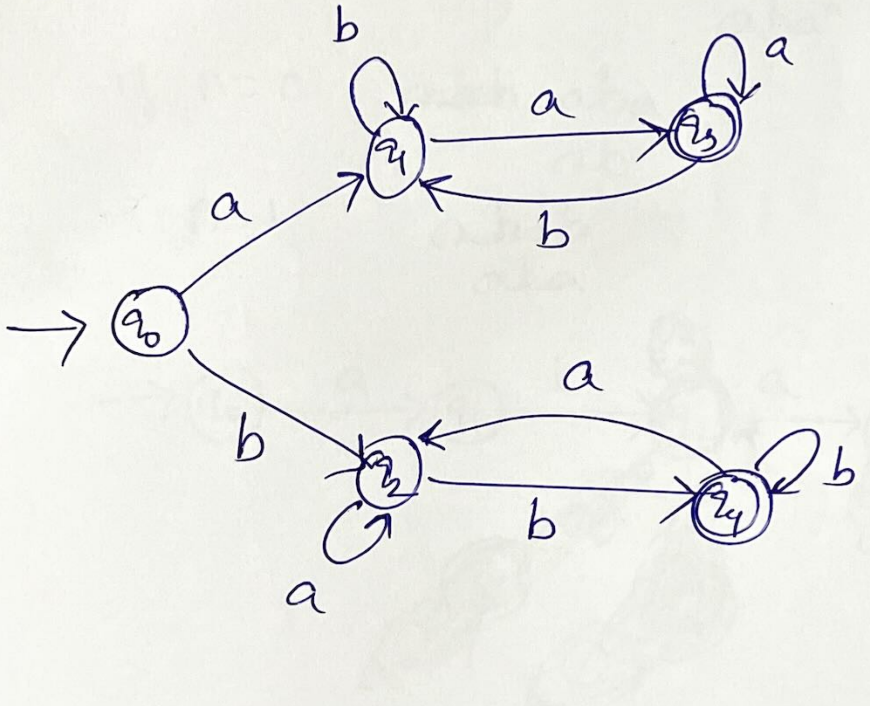
q_0
0
0
q_0
q_1
0
0
0
q_0
q_1
q_3
1
1
1
1
q_0
q_2
ϕ
q_3
1
1
1
1
q_0
q_2
q_3
q_3

Design a DFA and NFA with the same first and last symbols over Σ(a,b).

Ans:- **NFA:-**



DFA:- aba, bab

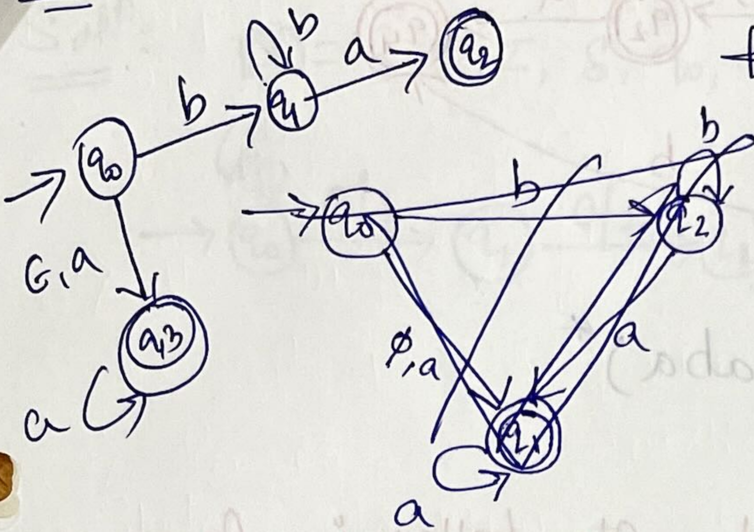


Design a NFA ~~with four state~~ for,

$L = \{a^n : n \geq 0\} \cup \{b^n a : n \geq 1\}$

Sol^n: Two cases are — $a^n : n \geq 0$

& $b^n a : n \geq 1$



Possible strings are:-

$a^n : n \geq 0 \Rightarrow \epsilon, a, aa, \ldots$

$b^n a : n \geq 1 \Rightarrow ba, bba, bbba \ldots$

$\epsilon, a, aa, ba, aaa, bba, \ldots$

Q4:- Design a NFA with no more than 5 States for the set : $L = \{abab^n : n \geq 0\} \cup \{aba^n : n \geq 0\}$.

Sol^n:- Two cases are:- $abab^n : n \geq 0$

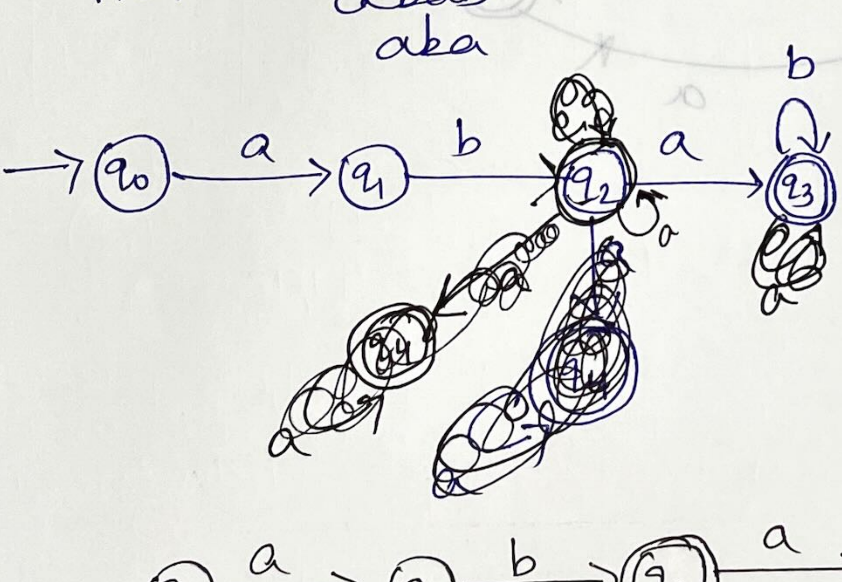
$aba^n : n \geq 0$

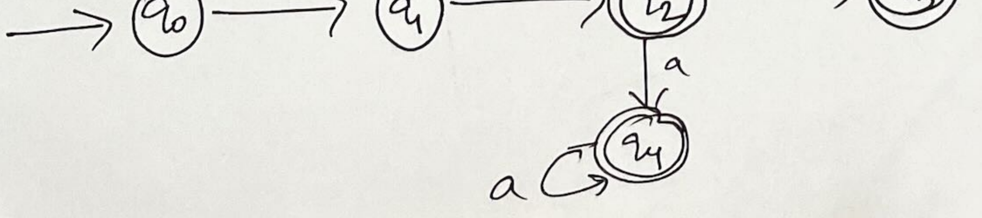
if $n = 0$ ~~aab~~ aba

ab

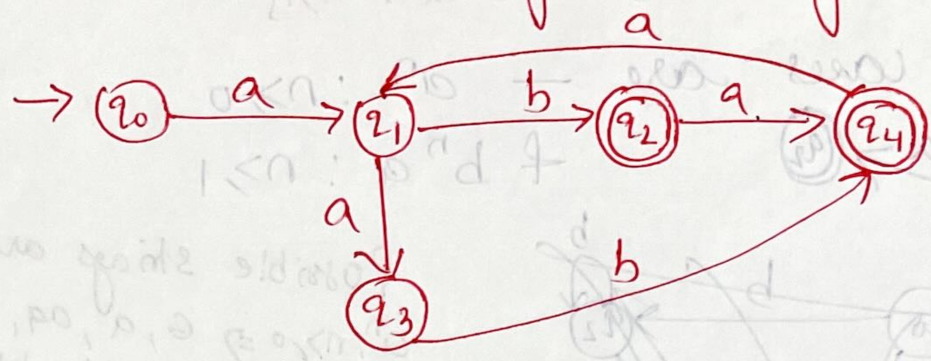
$n = 1$ abab

aba





Q: Find the language accepted by the D whose state diagram is given below.



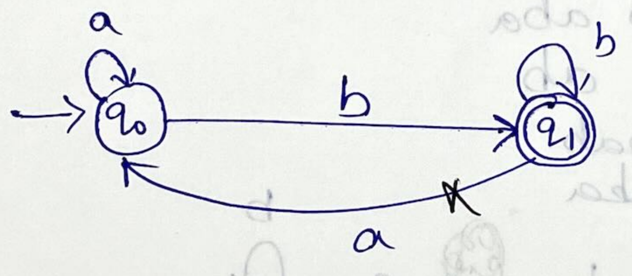
$$L = (ab + aab + aba)^*$$

Q: Construct the NFA for the following language:

$L = \{ x \in \{a,b\}^* : x$ containing any no. of a's followed by atleast one b.

aaab

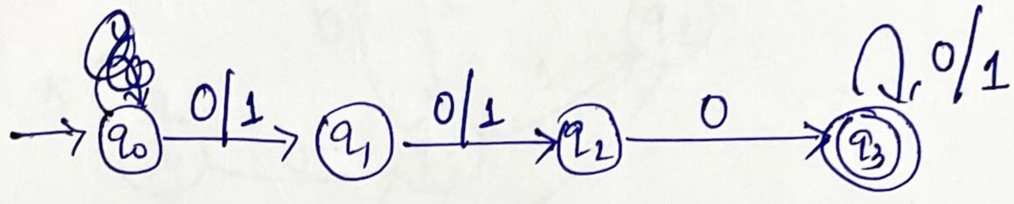
ab aab aaaab



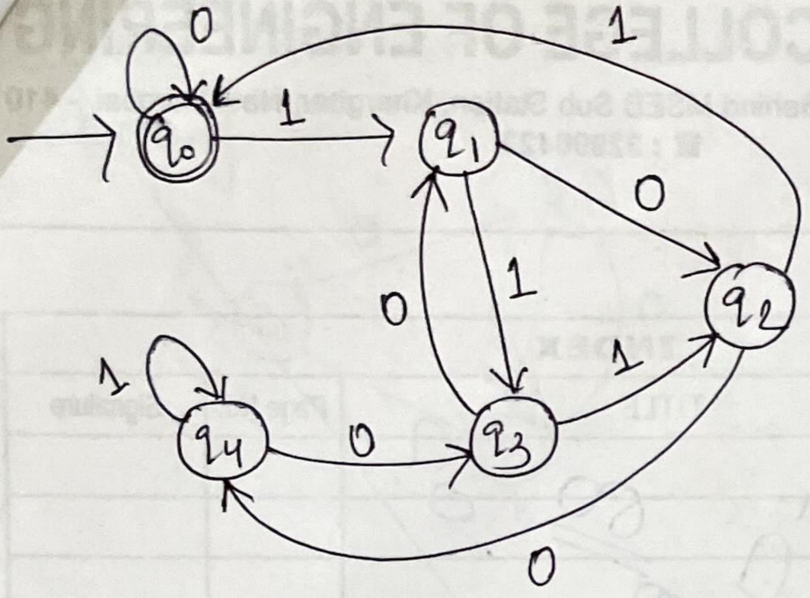
→ Draw NFA with four states for all strings over $\{0,1\}$ in which third symbol from right end is '0'.

Soln:

$M = \{Q, \Sigma, \delta, q_0, F\}$

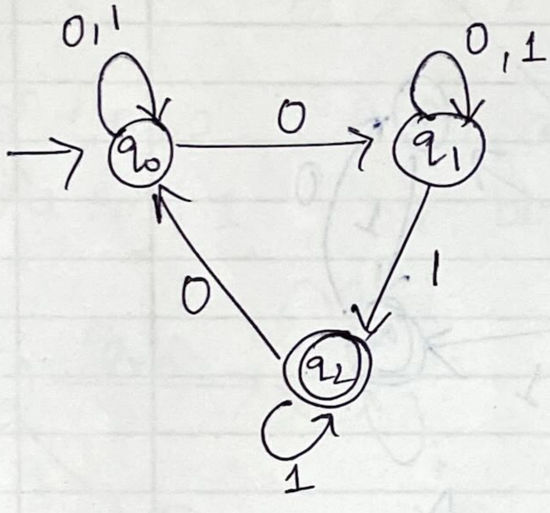


Construct the DFA for divisibility by 5.



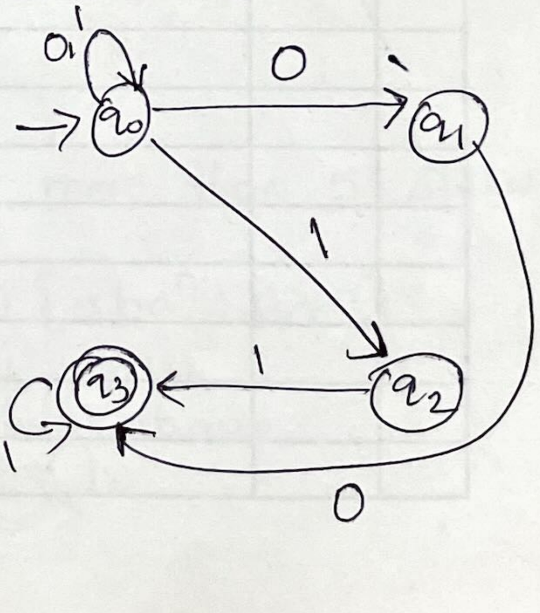
NFA

eg:-



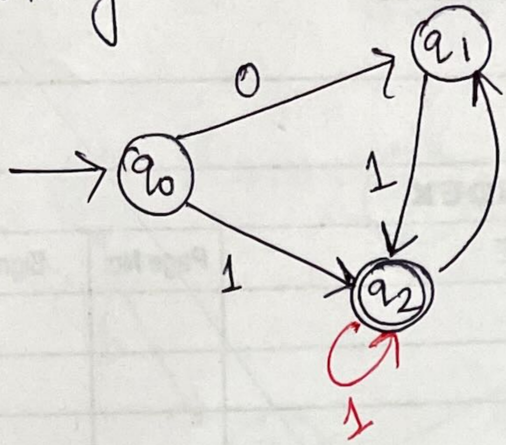
① Sketch the NFA for given State table

State	0	1
→ q_0	q_0, q_1	q_0, q_2
q_1	q_3	–
q_2	–	q_3
* q_3	q_3	q_3



Q: Determine NFA accepting all strings over {0, 1} which end in 1 but does not contain the substring 00.

Solⁿ:



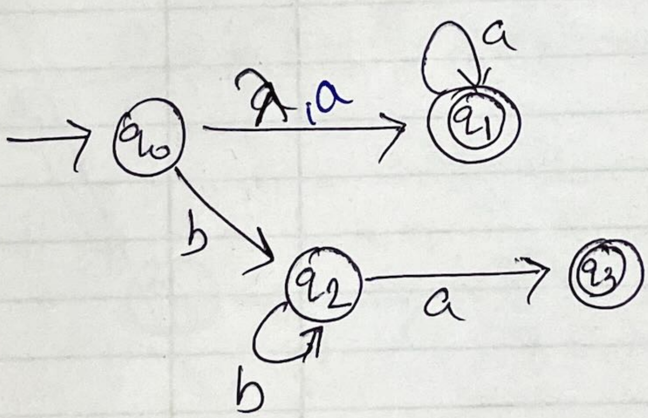
Q: Find an NFA with four states for

$L = \{a^n : n \geq 0\} \cup \{b^n a : n \geq 1\}$

Solⁿ:

i) $a^n : n \geq 0$, $\lambda, a, aa, aaa, \ldots$

ii) $b^n a : n \geq 1$, $ba, bba, bbba \ldots$

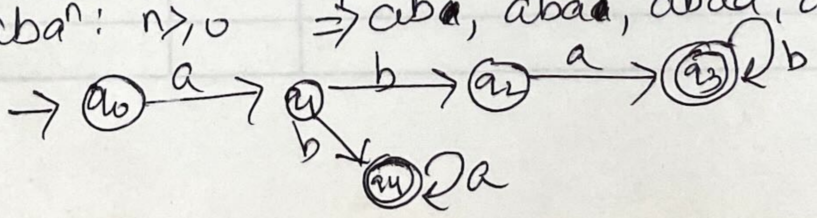


Q: Design an NFA with not more than 5 states for the set

$L = \{abab^n : n \geq 0\} \cup \{aba^n : n \geq 0\}$

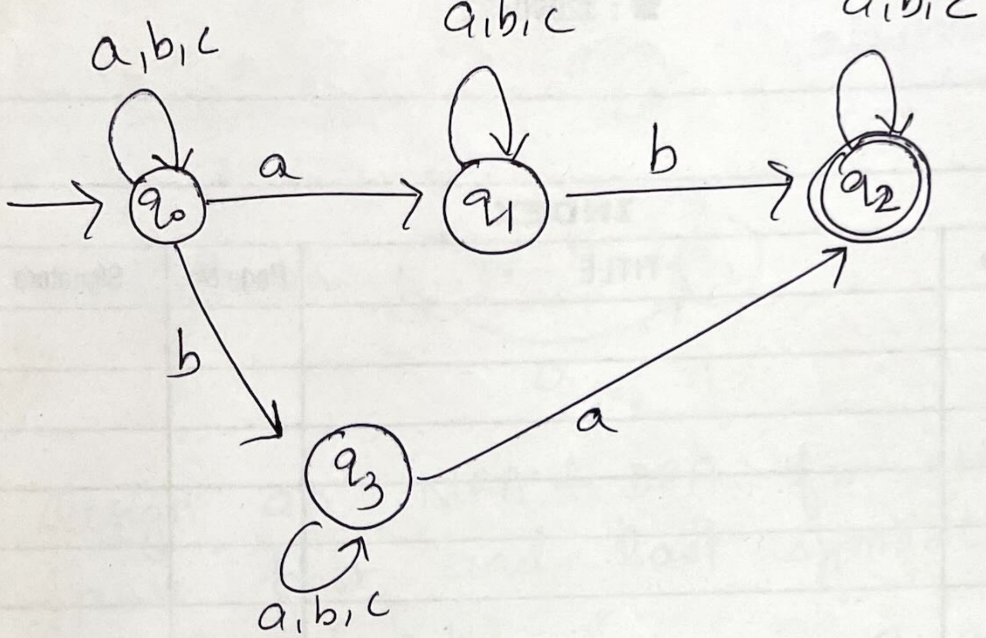
Solⁿ: $abab^n : n \geq 0 \Rightarrow aba, abab, ababb, ababbb \ldots$

$aba^n : n \geq 0 \Rightarrow aba, abaa, abaa, abaaa, \ldots$

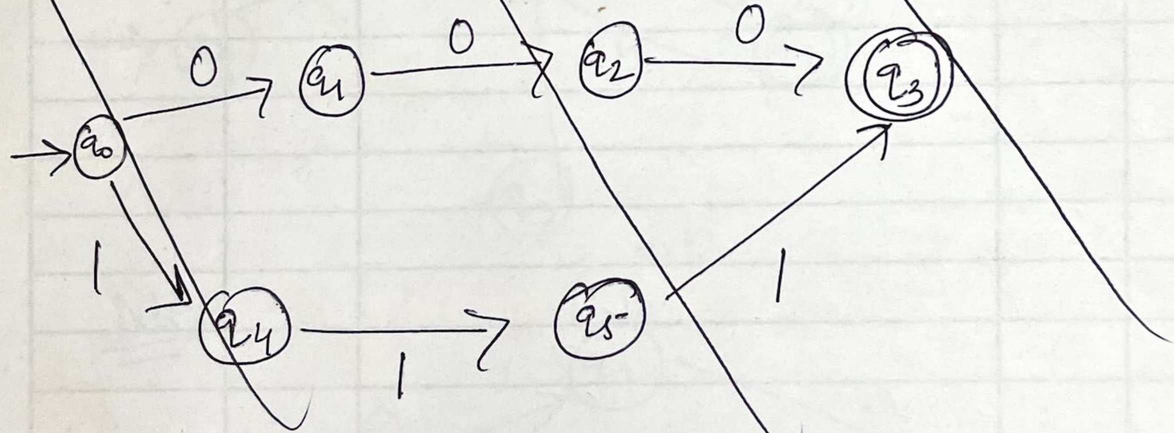


Design NFA.
Strings over {a, b, c} that contain atleast One a and atleast one b.

Sol^n:

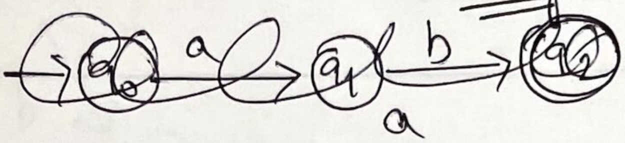


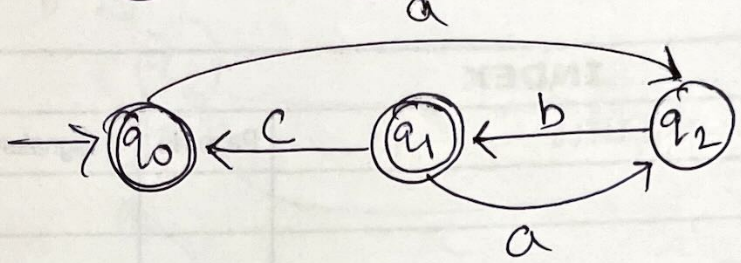
Q: Starts with 000 or ending with 111 (or both)



Design an NFA ~~for~~ with three states that accepts the language $\{ab, abc\}^*$

Strings: ϵ, ab, abc, ababc, abcab, abab, . . .

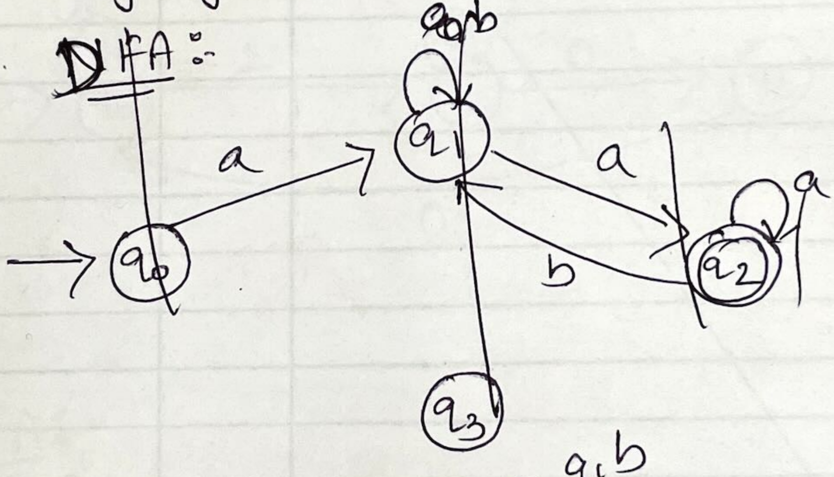




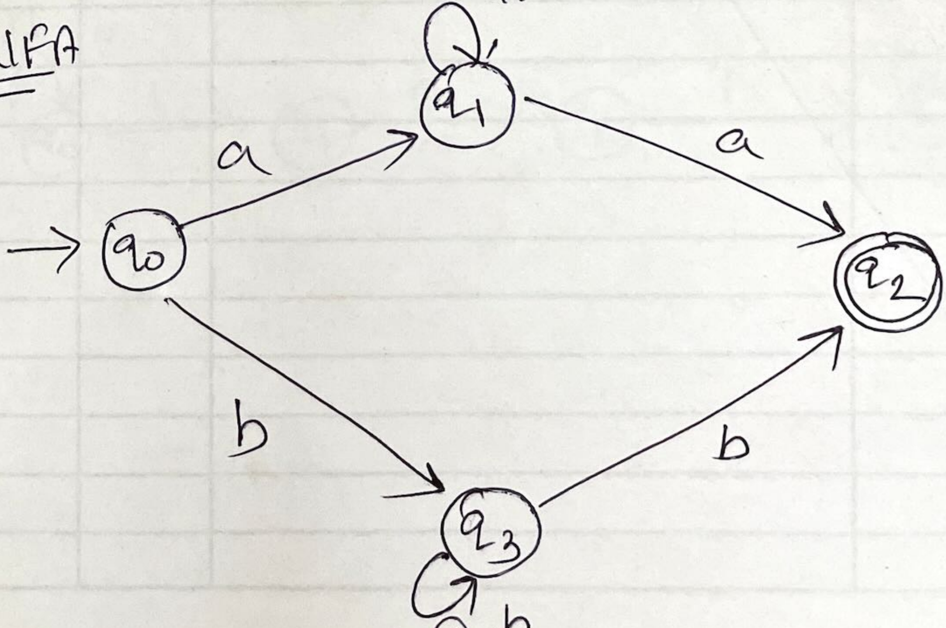
Q: Design an NFA & DFA for strings with the same first and last symbols over the language $\{a, b\}$.

a...a
b...b

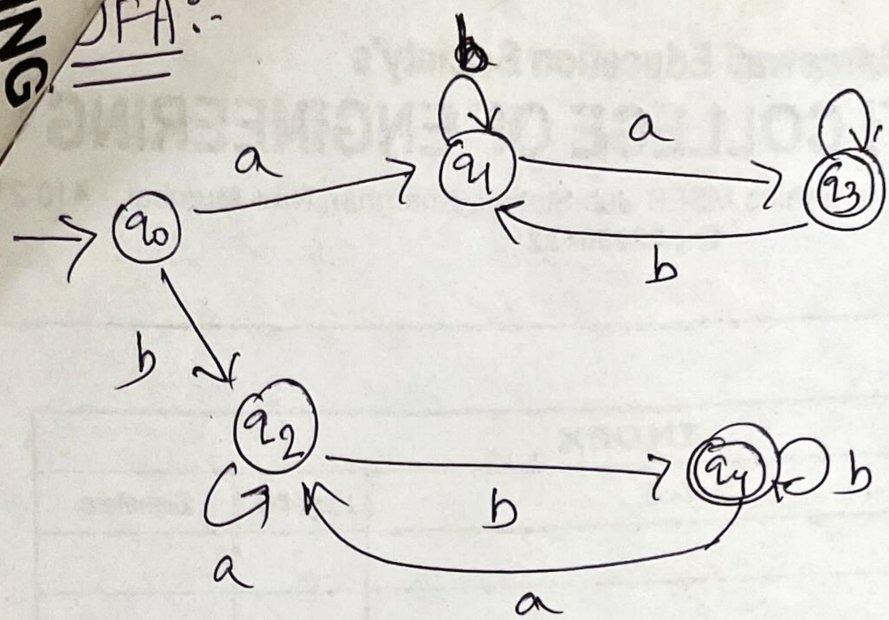
Sol^n: DFA:



NFA

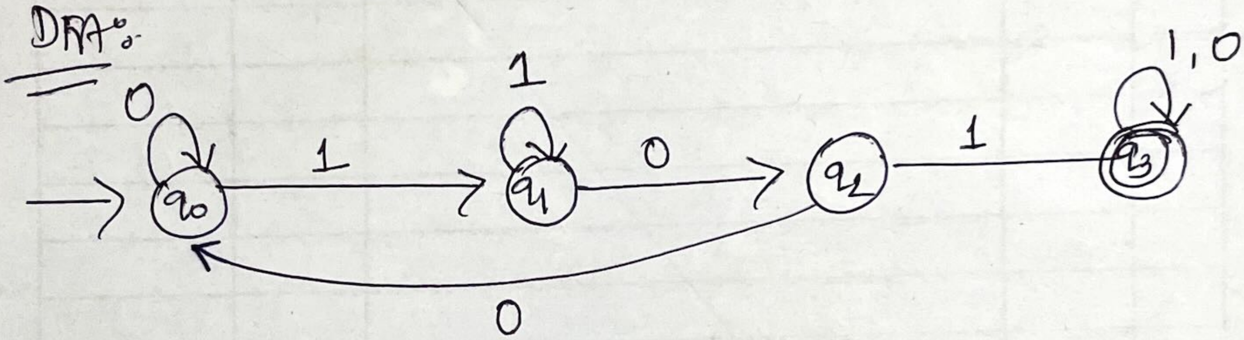


DFA:-

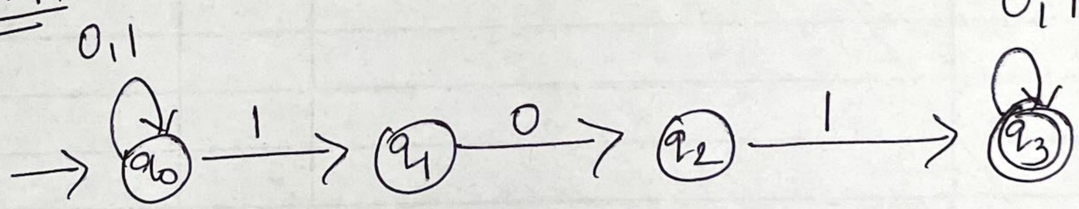


Q: Design an automan - NFA & DFA for searching the keyword 101 over language $\{0, 1\}$.

DFA:



NFA:



Equivalence of DFA and NDFA

1) A DFA can simulate the behavior of NFA by increasing the no. of states.

2) Any NFA is a more general machine without being more powerful.

Theorem:- For any NDFA or NFA, there exists a DFA which simulates the behavior of NFA i.e if L is the set of strings accepted by NFA, then there exists a DFA which also accepts L.

Construction of DFA equivalent to Given NFA

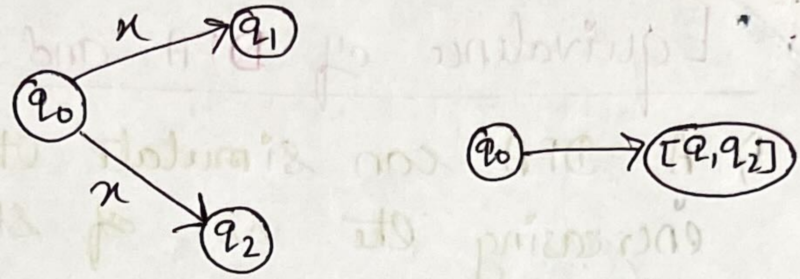
For every non deterministic FA, there exists an equivalent deterministic FA. Since the DFA equivalent of NFA is to simulate the moves of NFA in parallel every state of DFA will be combination of one or more states of NFA, hence every state of DFA will be represented by some subset of set of states of NFA and therefore, the transformation of NFA to DFA is normally called subset construction.

Thus, if a given NFA has n states, then the no. of states in the equivalent DFA will be 2^n.

8/11

[ ]

Single State



Algorithm:-

Conversion from NFA to equivalent DFA

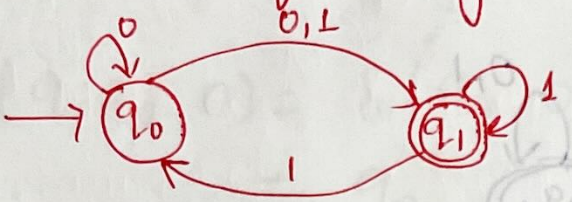
1) Seek all the transitions from the starting state q_0 for every symbol in Σ. If we get a set of states for some input then consider that set as a new single state.

2) In step(1) we are getting a new state then check all transitions of Σ for this new state only. Suppose new state as $[q_1, q_2]$ then for some input alphabet 'a' compute $\delta(q_1, a) \cup \delta(q_2, a)$. Let the result be $[q_1, q_m]$. Create a vertex $[q_1, q_m]$ as a new state if it does not already exist. Add an edge from $[q_1, q_2]$ to $[q_1, q_m]$ and label it with 'a'.

3) Repeat step 2 till we are getting new state.

4) All those states which consist of atleast one final state of given NFA as member state will be considered as final state.

Q: Convert the following NFA into DFA.



Solⁿ: Initially, we have the starting state q_0, so create start state as $[q_0]$.

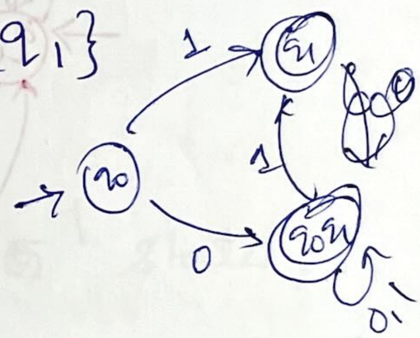
	0	1
$\to q_0$	$\{q_0, q_1\}$	q_1
$* q_1$	~~q_0q_1~~ -	q_0q_1
$* q_0q_1$	q_0q_1	q_0q_1

Now, $\delta([q_0], 0) = \{q_0, q_1\}$

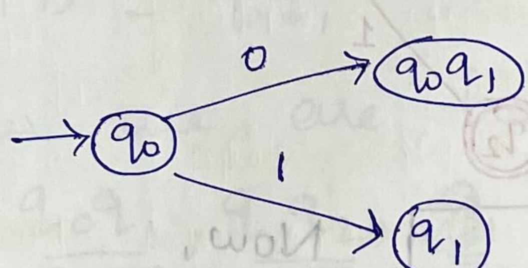
we will create new state as,

$\to [q_0] \xrightarrow{0} [q_0q_1]$

again we find $\delta([q_0], 1) = \{q_1\}$



i.e



Next for vertex q_0q_1, with input 0 & 1,

$\delta([q_0q_1], 0) = \delta(q_0, 0) \cup \delta(q_1, 0)$

$= q_0q_1 \cup \phi$

$= \{q_0q_1\}$

$\delta([q_0q_1], 1) = \delta(q_0, 1) \cup \delta(q_1, 1)$

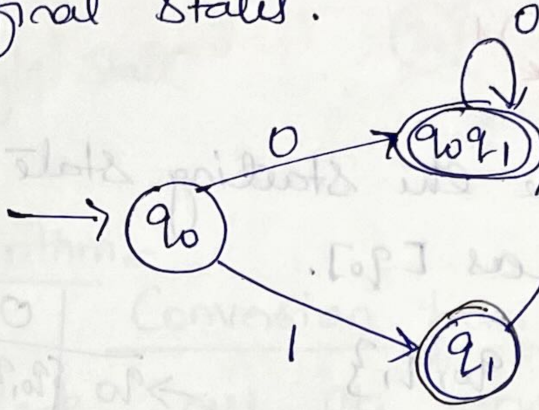
$= q_1 \cup \{q_0q_1\}$

$= \{q_0q_1\}$

Now for $\{q_1\}$

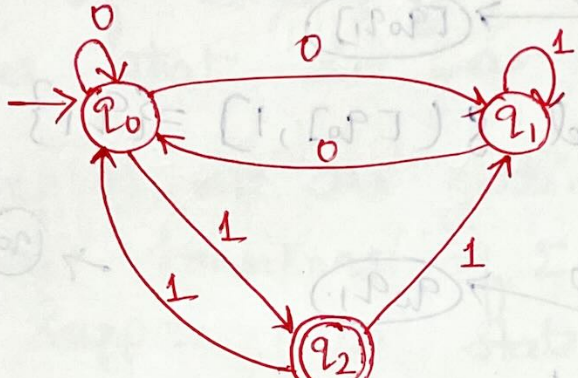
$\delta(q_1, 0) = \phi \quad \delta(q_1, 1) = q_0q_1$

Finally, the vertex $\{q_0q_1\}$ & $\{q_2\}$ are the final states.



— Equivalent DFA

Q: Convert the following NFA to its equivalent DFA.



Solⁿ:

State	I/P 0	I/P 1
q_0	q_0q_1	q_2
q_1	q_0	q_1
$*q_2$	—	q_0q_1
q_0q_1	q_0q_1	q_1q_2
q_1q_2	q_0	q_0q_1

Now,

$\delta(q_0, 0) = q_0q_1$

$\delta(q_0, 1) = q_2$

Consider q_0q_1 as a new state.

$\delta(q_0q_1, 0) = \delta(q_0,0) \cup \delta(q_1,0)$
$= q_0q_1 \cup q_0$
$= q_0q_1$

$\delta(q_0q_1, 1) = \delta(q_0,1) \cup \delta(q_1,1)$
$= q_2 \cup q_1$
$= q_1q_2$

We got new state q_1q_2.
Create new state q_1q_2.

$$\delta(q_1q_2, 0) = \delta(q_1, 0) \cup \delta(q_2, 0)$$
$$= \underline{q_0}$$

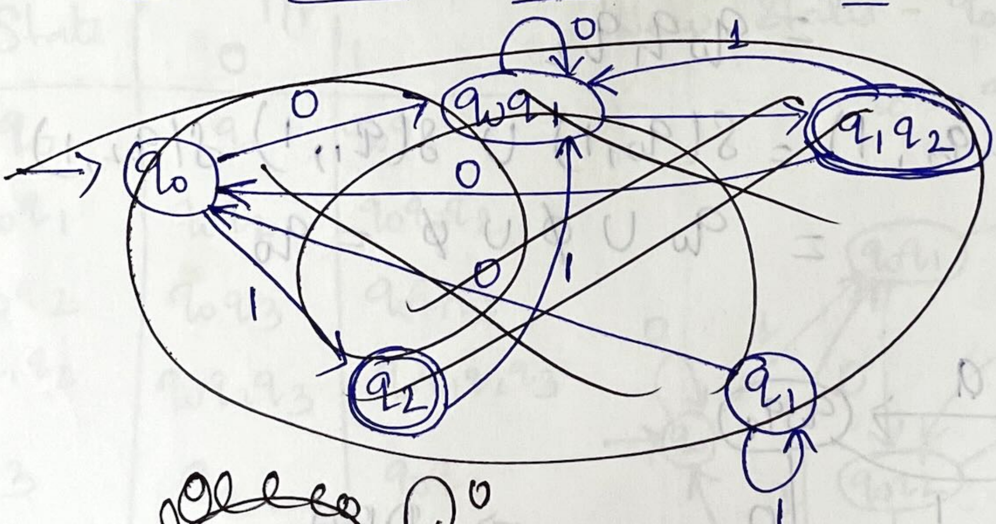
$$\delta(q_1q_2, 1) = \delta(q_1, 1) \cup \delta(q_2, 1)$$
$$= \underline{q_0q_1}$$

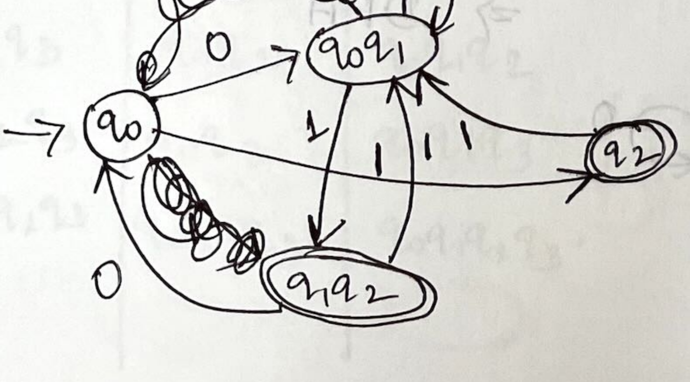
$$\times \left\{ \begin{array}{l} \delta(q_1, 0) = q_0 \\ \delta(q_1, 1) = q_1 \end{array} \right\}$$

$$\delta(q_2, 0) = -$$
$$\delta(q_2, 1) = q_0q_1$$

So, now we are having ~~5~~ 4 states,

$\underline{q_0}$, $\underline{q_0q_1}$, $\underline{q_1q_2}$, $\underline{q_1}$, $\underline{q_2}$.

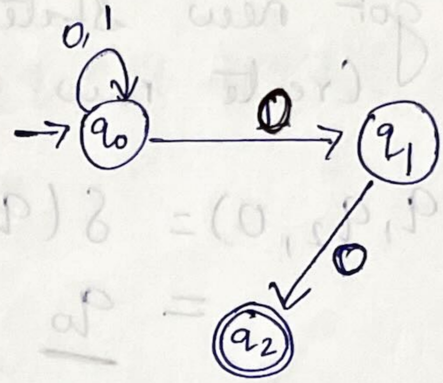




Equivalent DFA

Q :-

State	i/p 0	i/p 1
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	–
$*q_2$	–	–



$\delta(q_0, 0) = q_0q_1$

$\delta(q_0, 1) = q_0$

New state q_0q_1

$\delta(q_0q_1, 0) = \delta(q_0, 0) \cup \delta(q_1, 0)$

$= q_0q_1 \cup q_2 = q_0q_1q_2$

$\delta(q_0q_1, 1) = \delta(q_0, 1) \cup \delta(q_1, 1)$

$= q_0 \cup \phi = \underline{q_0}$

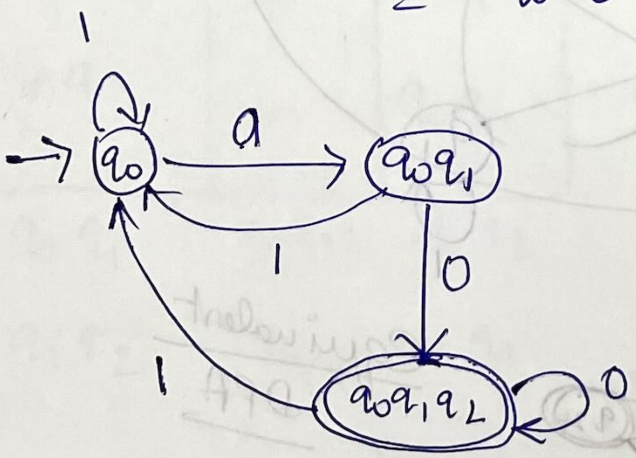
$\delta(q_0q_1q_2, 0) = \delta(q_0, 0) \cup \delta(q_1, 0) \cup \delta(q_2, 0)$

$= q_0q_1 \cup q_2 \cup \phi$

$= q_0q_1q_2$

$\delta(q_0q_1q_2, 1) = \delta(q_0, 1) \cup \delta(q_1, 1)\ \delta(q_2, 1)$

$= q_0 \cup \phi \cup \phi = q_0$



$\Rightarrow$ DFA

Q:- $M = \{Q, \Sigma, q_0, \delta, F)$

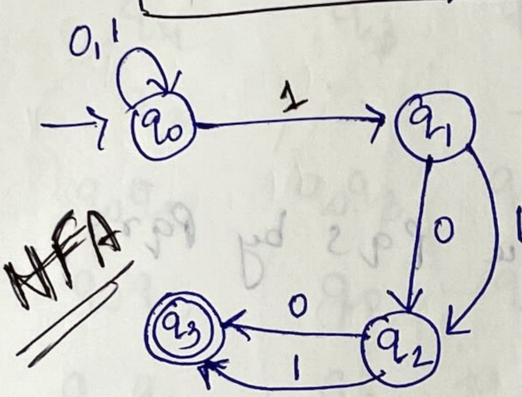
$Q = (q_0, q_1, q_2, q_3)$

$F = q_3$

$\Sigma = 0, 1$

Given NFA

State	i/p 0	1
$\rightarrow q_0$	q_0	q_0q_1
q_1	q_2	q_2
q_2	q_3	q_3
$*q_3$	ϕ	ϕ



NFA

$\delta(q_0, 0) = q_0$

$\delta(q_0, 1) = q_0q_1$

New State - q_0q_1

$\delta(q_0q_1, 0) = q_0q_2$

$\delta(q_0q_1, 1) = q_0q_1q_2$

New State - q_0q_2 & $q_0q_1q_2$

$\delta(q_0q_2, 0) = q_0q_3$

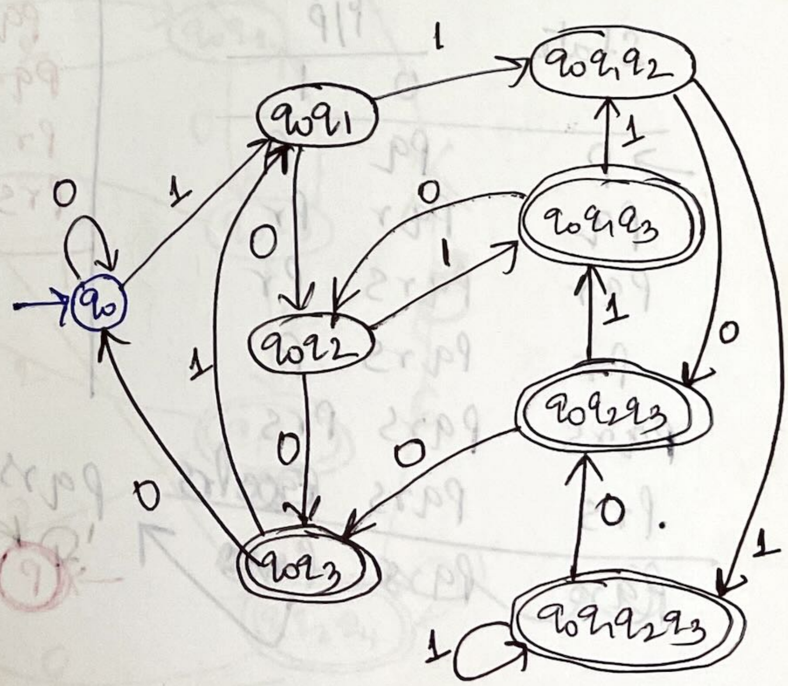
$\delta(q_0q_2, 1) = q_0q_1q_3$

$\delta(q_0q_1q_2, 0) = q_0q_2q_3$

$\delta(q_0q_1q_2, 1) = q_0q_1q_2q_3$

New States - q_0q_3, $q_0q_1q_3$, $q_0q_2q_3$, $q_0q_1q_2q_3$

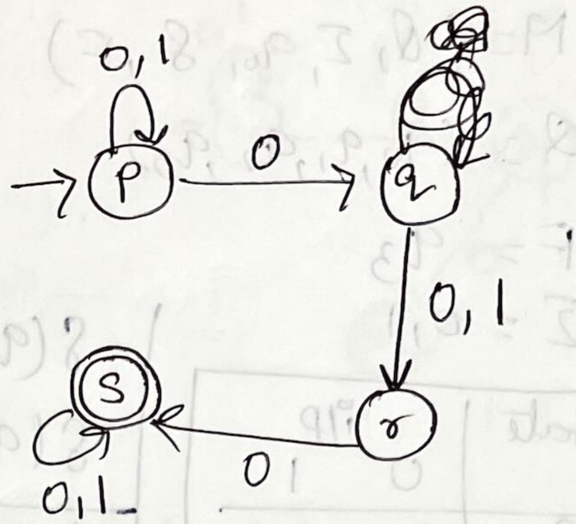
State	i/p 0	1
q_0	q_0	q_0q_1
q_0q_1	q_0q_2	$q_0q_1q_2$
q_0q_2	q_0q_3	$q_0q_1q_3$
$q_0q_1q_2$	$q_0q_2q_3$	$q_0q_1q_2q_3$
$*q_0q_3$	q_0	q_0q_1
$*q_0q_1q_3$	q_0q_2	$q_0q_1q_2$
$*q_0q_2q_3$	q_0q_3	$q_0q_1q_3$
$*q_0q_1q_2q_3$	$q_0q_2q_3$	$q_0q_1q_2q_3$



8||

Q:

	0	1
→p	pq	p
q	r	r
r	s	-
*s	s	s



State	i/p 0	1
→p	pq	p
pq	pqr	pr
pqr	pqrs	pr
pr	pqs	p
*pqrs	pqrs	prs
*pqs	pqrs	prs
*prs	pqs	ps
*ps	pqs	ps

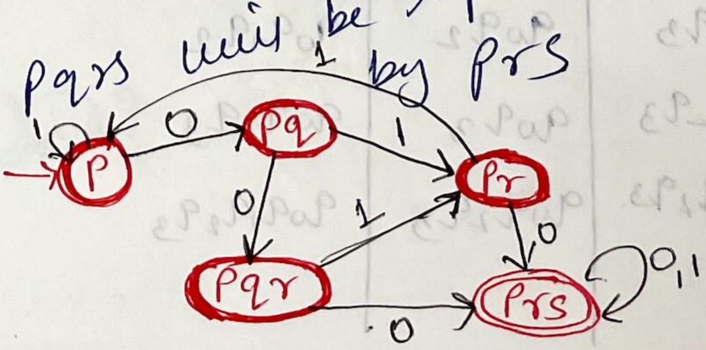
Replace pqs by pqrs

Replace ps by prs

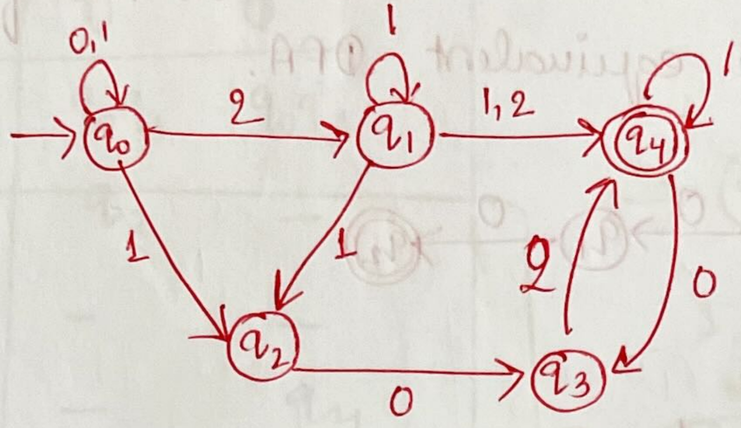
State	i/p 0	1
→p	pq	p
pq	pqr	pr
pqr	pqrs	pr
pr	pqrs	p
pqrs	pqrs	prs
.prs	pqrs	~~pqrs~~ prs
~~pqrs~~	~~pqrs~~	~~pqrs~~

pqrs will be replaced by prs

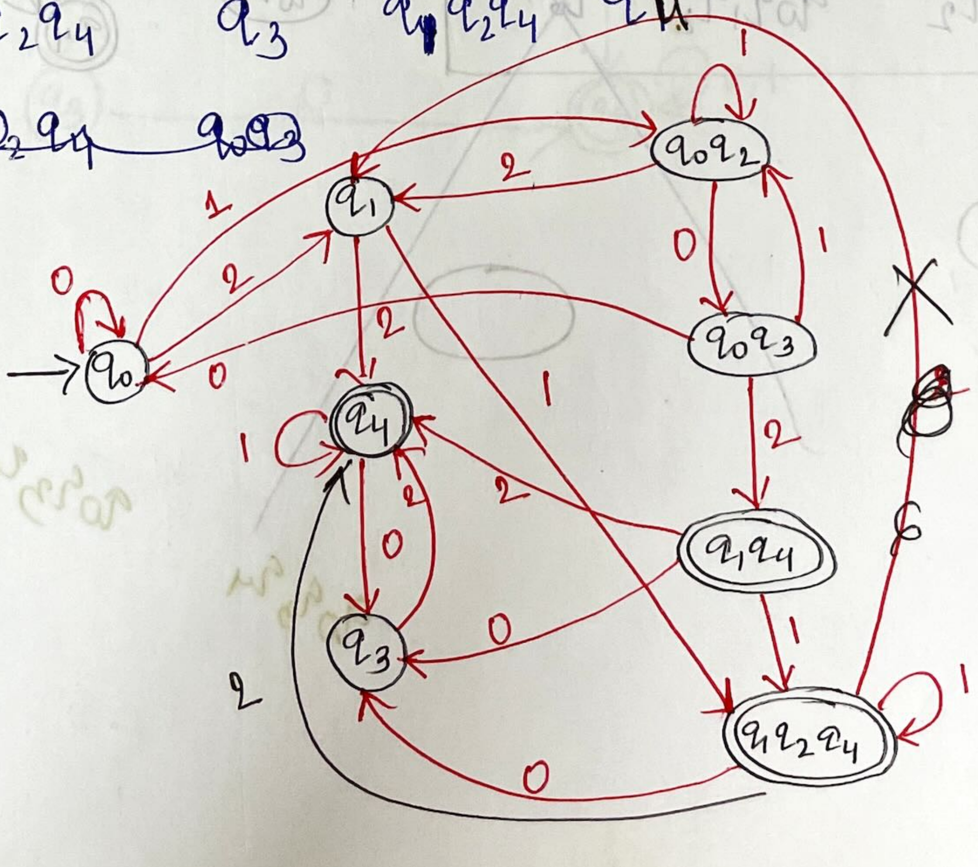
	0	1
→p	pq	p
pq	pqr	pr
pqr	prs	pr
pr	prs	p
prs	prs	prs



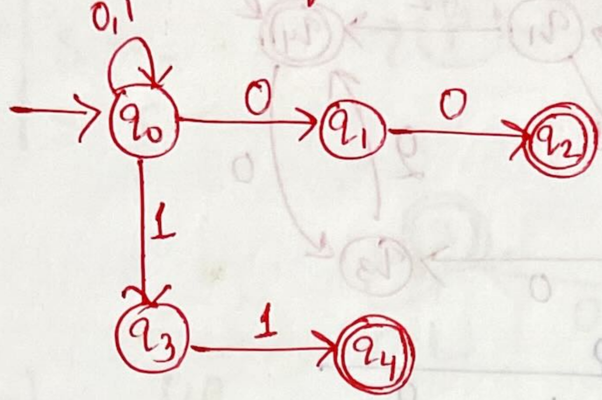
Q: Construct DFA for given NFA



State	i/p 0	1	2
$\rightarrow q_0$	q_0	q_0q_2	q_1
q_1	$\varnothing$	$q_1q_2q_4$	q_4
q_4	q_3	q_4	$\varnothing$
q_3	—	—	q_4
q_0q_2	q_0q_3	q_0q_2	q_1
q_0q_3	q_0	q_0q_2	q_1q_4
q_1q_4	q_3	$q_1q_2q_4$	q_4
$q_1q_2q_4$	q_3	$q_1q_2q_4$	q_4
~~$q_0q_2q_4$~~	~~q_0q_3~~		

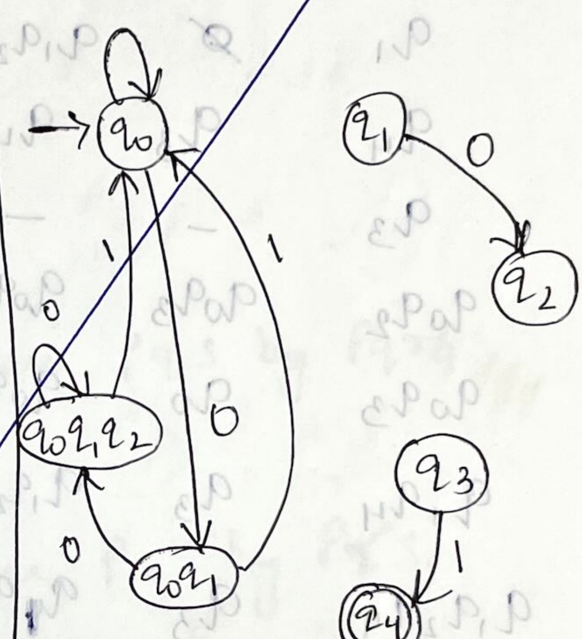


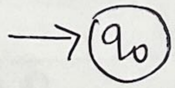
Q :- Given the NFA as shown in the figure, determine the equivalent DFA.

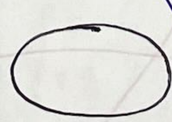


Soln :-

State	i/p 0	1
$\rightarrow q_0$	q_0q_1	q_0
q_1	q_2	—
q_2	—	—
q_3	—	q_4
$* q_4$	—	—
q_0q_1	$q_0q_1q_2$	q_0
$q_0q_1q_2$	$q_0q_1q_2$	q_0



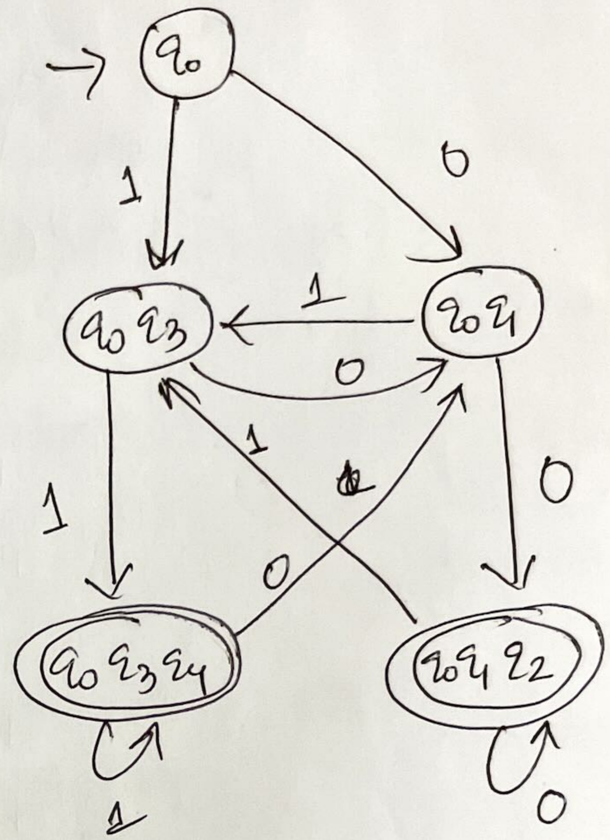




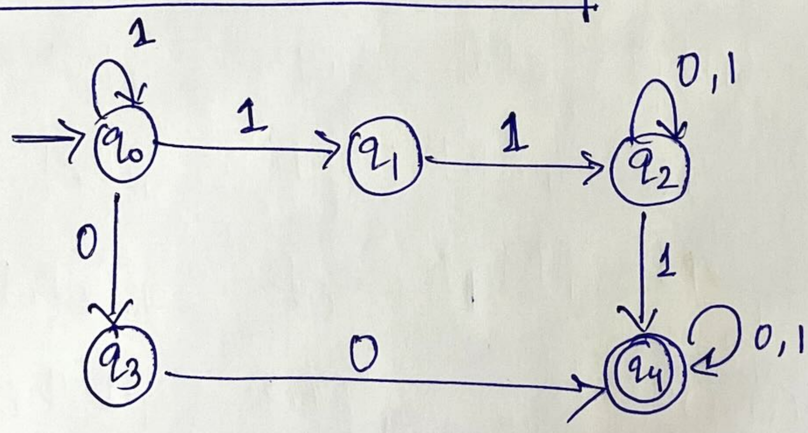
$q_0q_3q_4$

$q_0q_3q_4$

State	i/p 0	1
$\rightarrow q_0$	q_0q_1	q_0q_3
q_1	q_2	–
q_2	–	–
q_3	–	q_4
$*q_4$	–	–
q_0q_1	$q_0q_1q_2$	q_0q_3
q_0q_3	q_0q_1	$q_0q_3q_4$
$*q_0q_1q_2$	$q_0q_1q_2$	q_0q_3
$*q_0q_3q_4$	q_0q_1	$q_0q_3q_4$

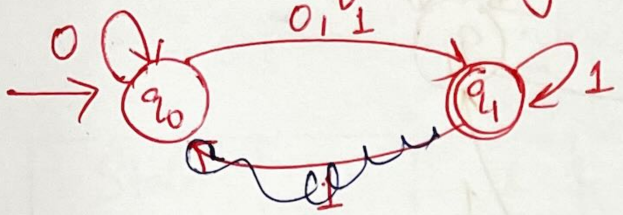


Q:



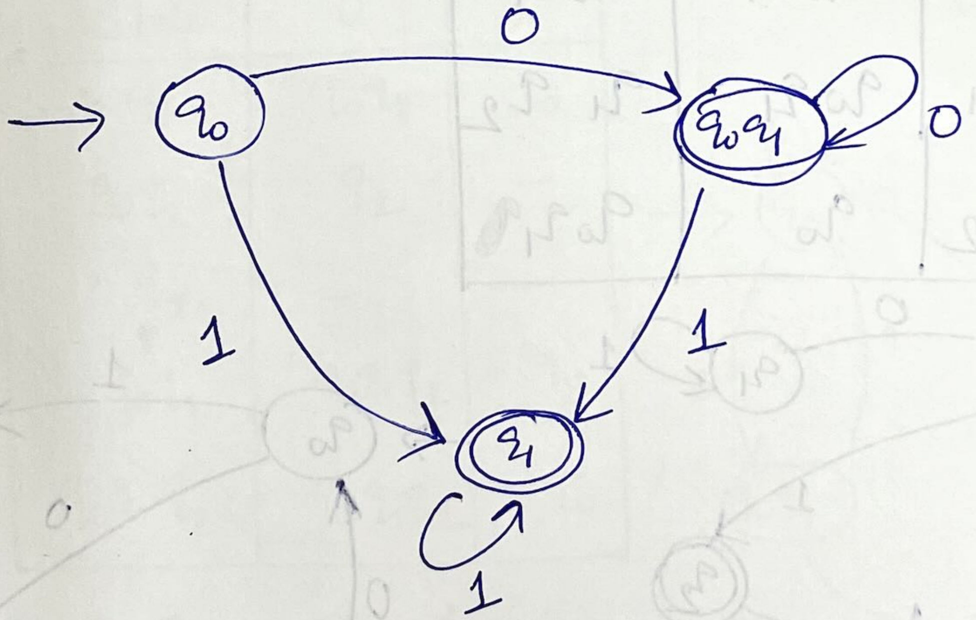
Conversion from NFA to DFA

Q:- Convert the following NFA to DFA.



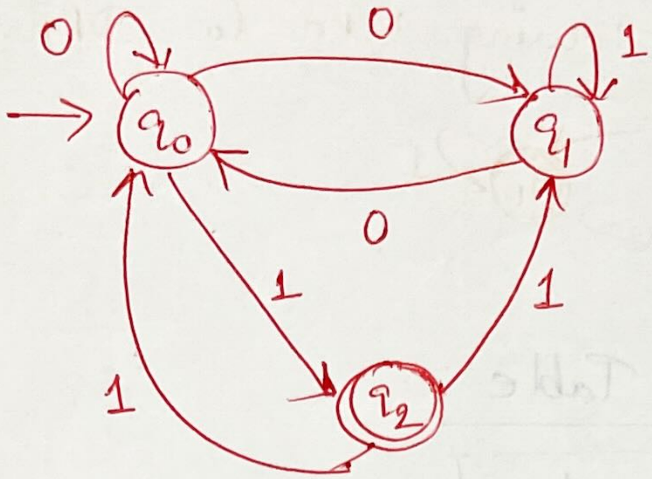
Solⁿ: Transition Table

	0	1
$\rightarrow q_0$	$q_0 q_1$	q_1
$* q_1$	—	q_1
$* q_0 q_1$	$q_0 q_1$	q_1



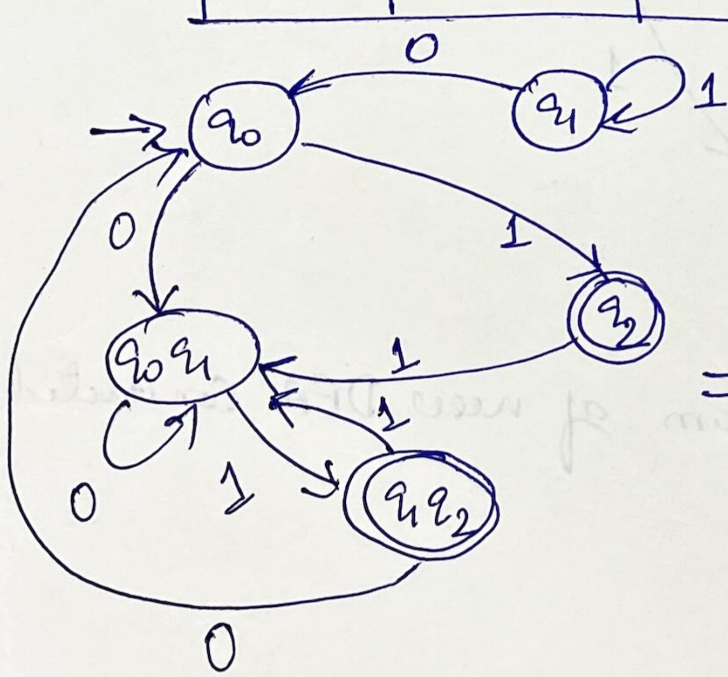
Transition diagram of new DFA constructed from NFA.

Q: Convert the following NFA to its equivalent DFA.

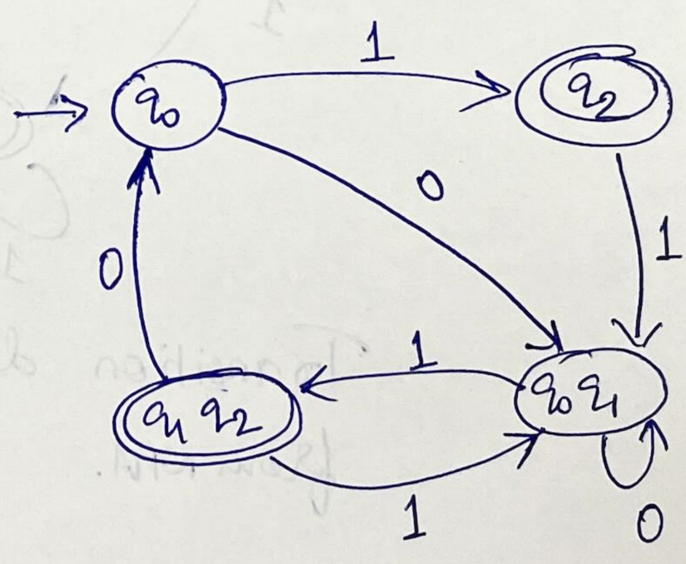


Solⁿ: Transition Table

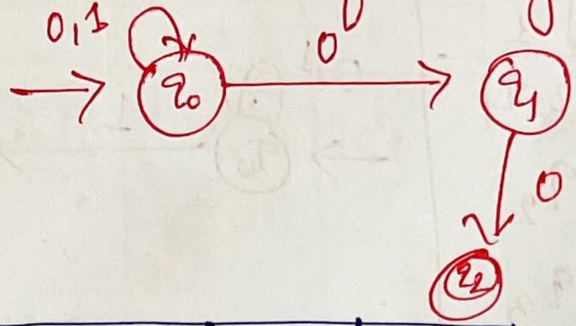
	0	1
$\rightarrow q_0$	q_0q_1	q_2
q_1	q_0	q_1
$*q_2$	q_0q_1	q_0q_1
q_0q_1	q_0q_1	q_1q_2
$*q_1q_2$	q_0	q_0q_1



$\Rightarrow$



Q:- Convert the following NFA to DFA.



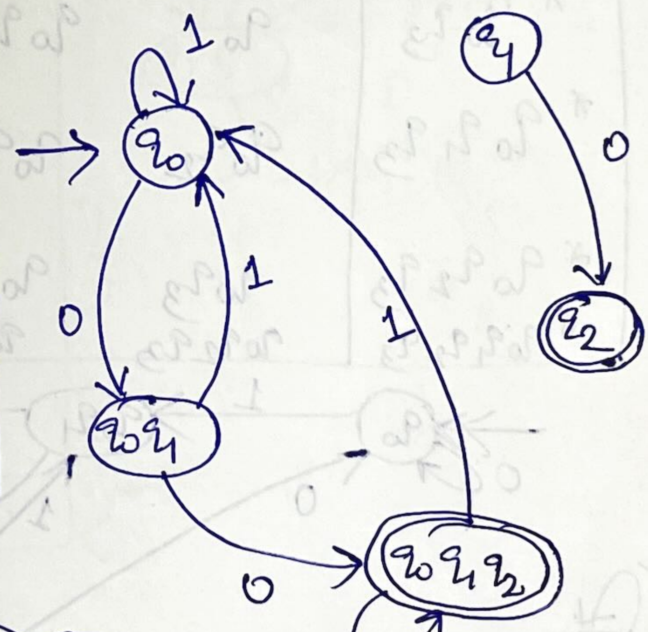
Sol^n:-

State	0	1
$\rightarrow q_0$	$q_0 q_1$	q_0
q_1	q_2	—
$* q_2$	—	—

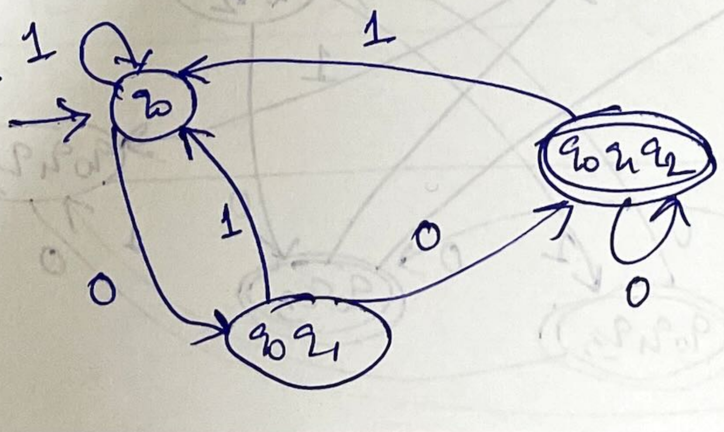
New transition table for DFA

we have to add new states, till we get the new one.

State	0	1
$\rightarrow q_0$	$q_0 q_1$	q_0
q_1	q_2	—
$* q_2$	—	—
$q_0 q_1$	$q_0 q_1 q_2$	q_0
$q_0 q_1 q_2$	$q_0 q_1 q_2$	q_0

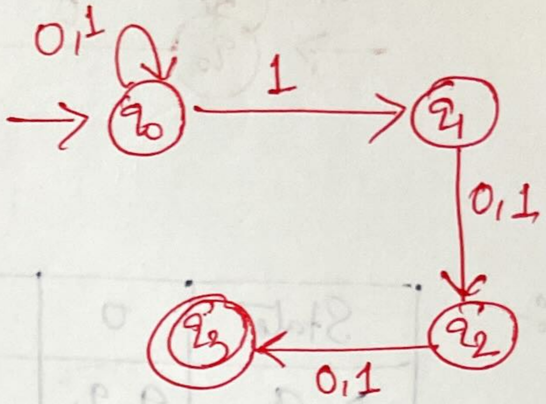


DFA:-

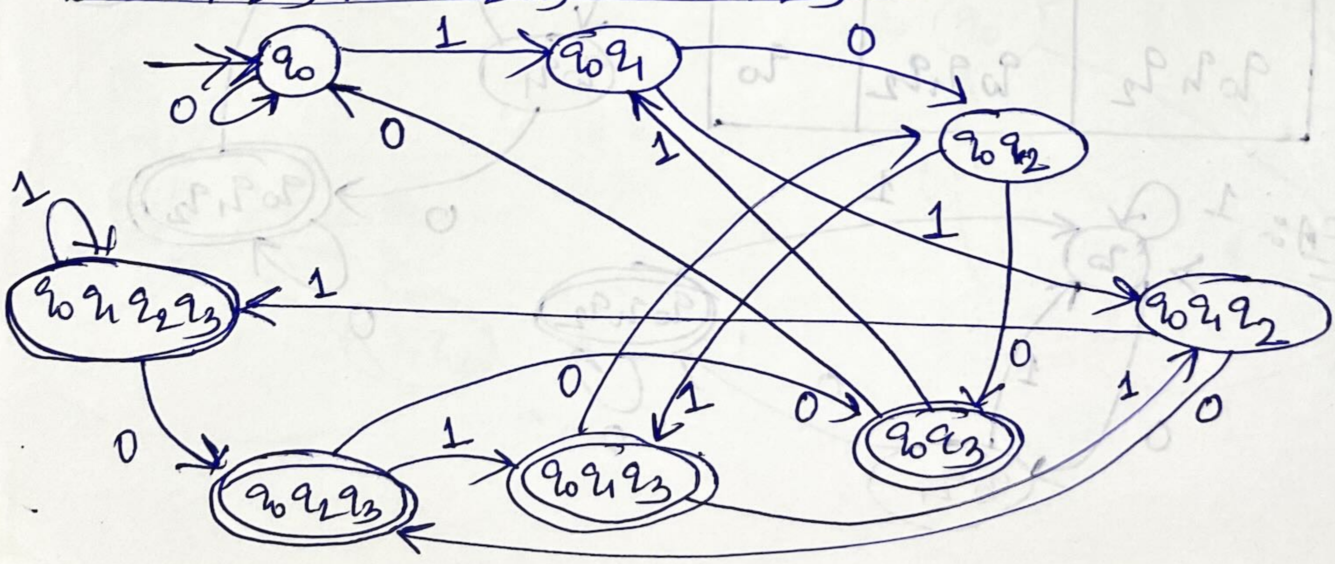


Q:- Convert the given NFA to DFA:-

	0	1
$\rightarrow q_0$	q_0	$q_0 q_1$
q_1	q_2	q_2
q_2	q_3	q_3
$*q_3$	ϕ	ϕ

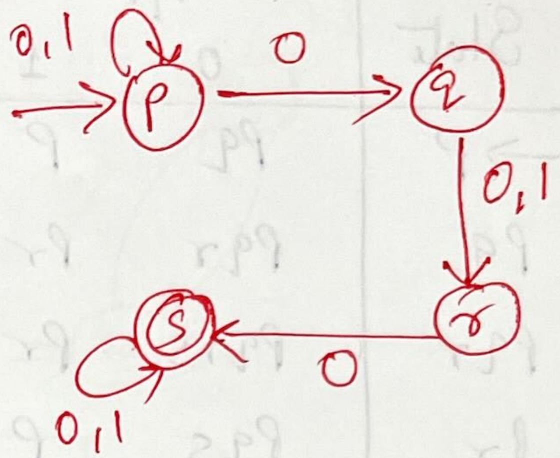


State	0	1
$\rightarrow q_0$	q_0	$q_0 q_1$
$q_0 q_1$	$q_0 q_2$	$q_0 q_1 q_2$
$q_0 q_2$	$q_0 q_3$	$q_0 q_1 q_3$
$q_0 q_1 q_2$	$q_0 q_2 q_3$	$q_0 q_1 q_2 q_3$
$*q_0 q_3$	q_0	$q_0 q_1$
$*q_0 q_1 q_3$	$q_0 q_2$	$q_0 q_1 q_2$
$*q_0 q_2 q_3$	$q_0 q_3$	$q_0 q_1 q_3$
$*q_0 q_1 q_2 q_3$	$q_0 q_2 q_3$	$q_0 q_1 q_2 q_3$



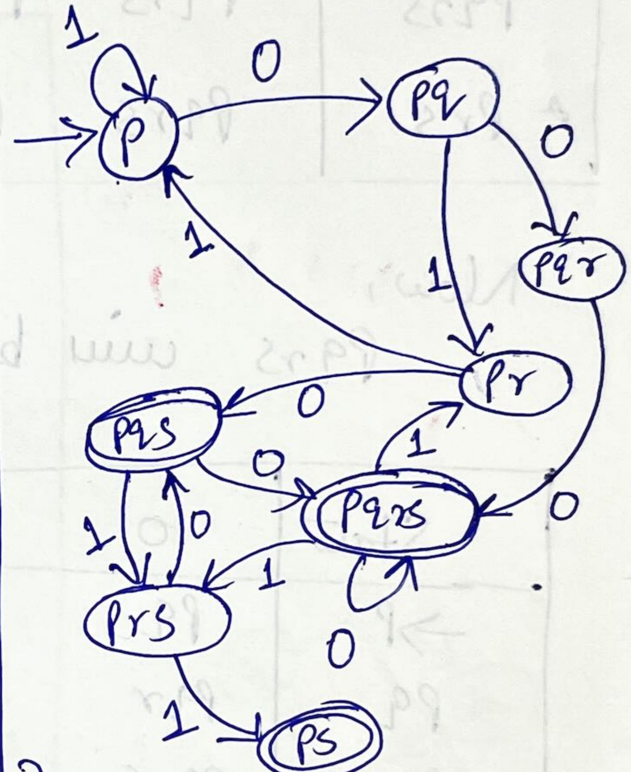
Q: Convert NFA to DFA

	0	1
→ p	pq	p
q	r	r
r	s	–
* s	s	s



Solⁿ:-

State	0	1
→ p	pq	p
q	r	r
r	s	–
* s	s	s
pq	pqr	pr
pqr	pqrs	pr
pr	pqs	p
* pqrs	pqrs	prs
* pqs	pqrs	prs
* prs	pqs	ps
* ps	pqs	ps



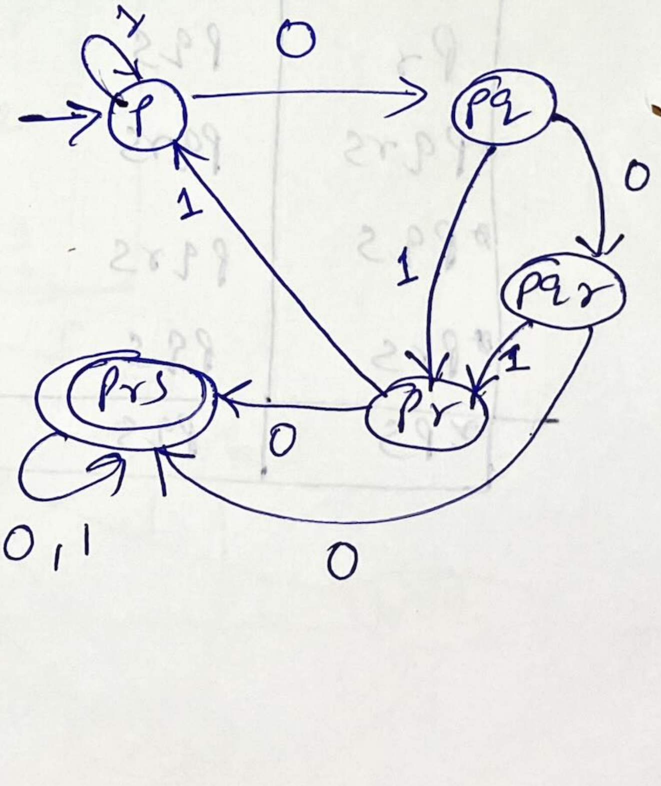
State	i/p 0	1
→p	pq	p
pq	pqr	pr
pqr	pqrs	pr
pr	pqs	p
* pqrs	pqrs	prs
* prs	pqrs	prs

pqs is substituted with pqrs.

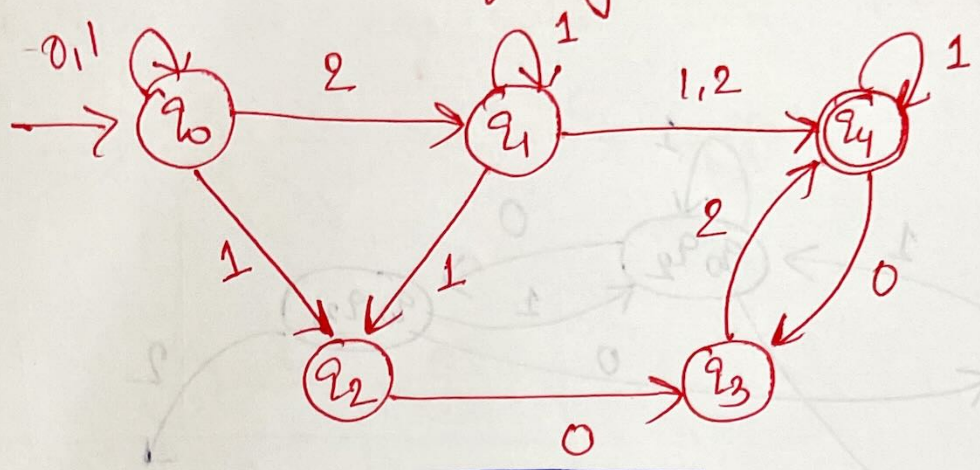
ps is substituted with prs.

Now,
pqrs will be substituted with prs.

State	0	1
→p	pq	p
pq	pqr	pr
pqr	pqrs	pr
pr	prs	p
* prs	prs	prs

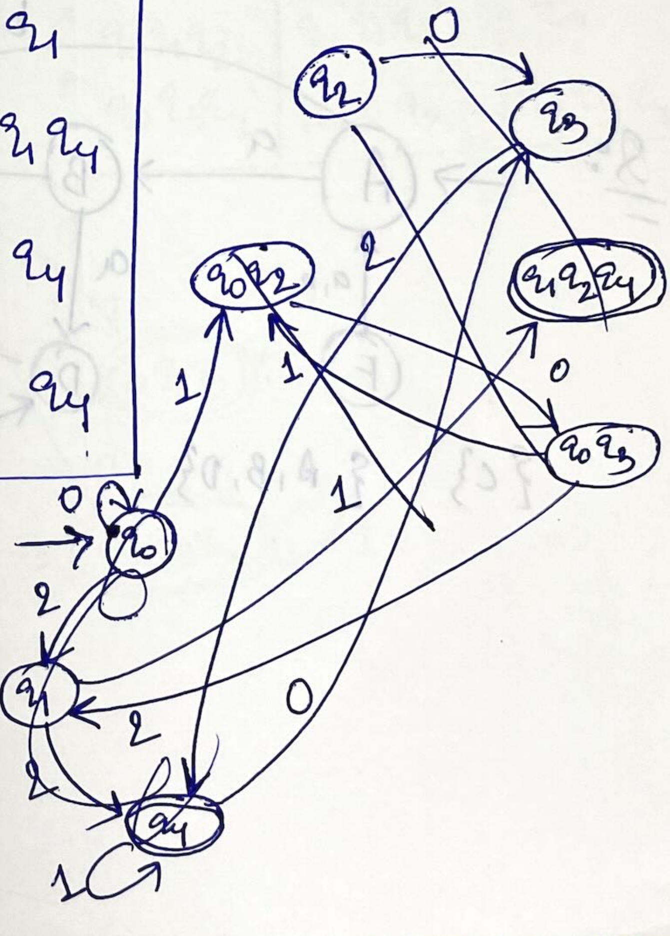


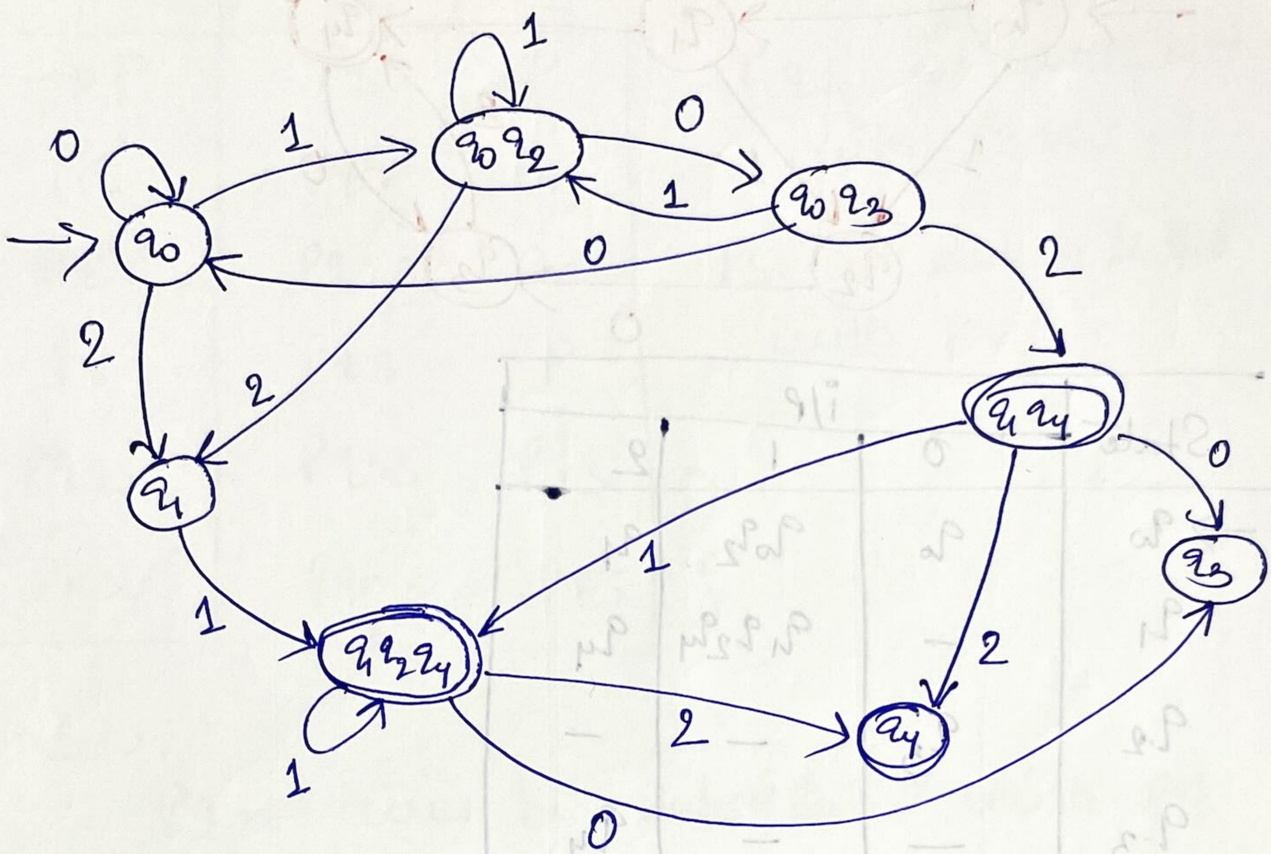
Q: Construct DFA for given NFA.



Solⁿ:-

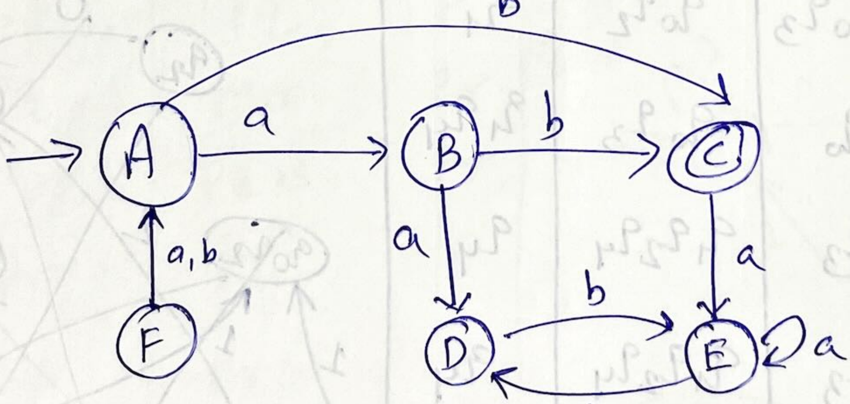
State	i/p 0	i/p 1	i/p 2
→ q_0	q_0	q_0q_2	q_1
q_1	—	$q_1q_2q_4$	q_4
q_2	q_3	—	—
q_3	—	—	q_4
* q_4	q_3	q_4	—
q_0q_2	q_0q_3	q_0q_2	q_1
q_0q_3	q_0	q_0q_3	q_1q_4
* q_1q_4	q_3	$q_1q_2q_4$	q_4
* $q_1q_2q_4$	q_3	$q_1q_2q_4$	q_4





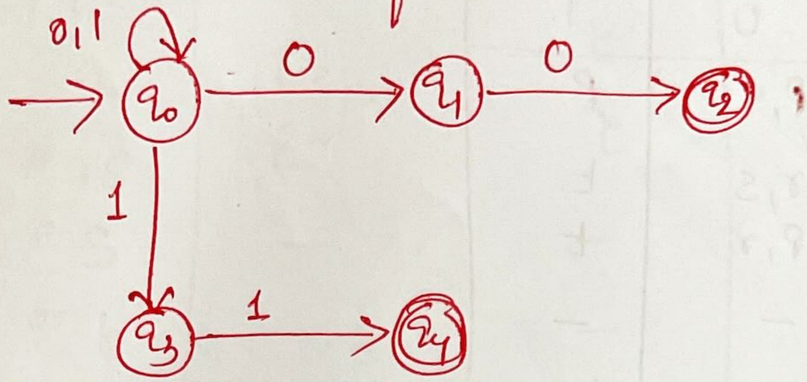
DFA of Converted NFA

Q:



{C}   {A, B, D}

Given the NFA as shown in the figure, determine the equivalent DFA.

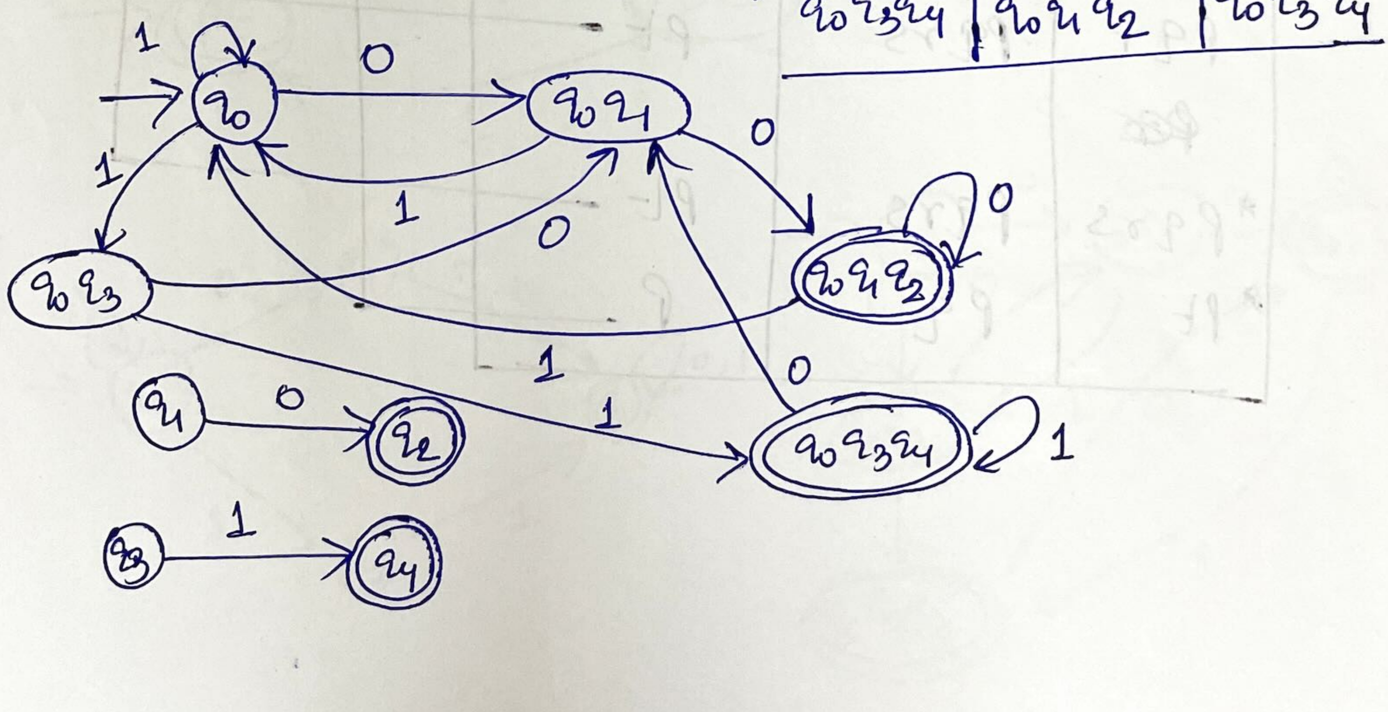


Solⁿ:

	0	1
$\rightarrow q_0$	q_0q_1	q_0q_3
q_1	q_2	—
q_3	—	q_4
$*q_2$	—	—
$*q_4$	—	—
q_0q_1	$q_0q_1q_2$	q_0q_3
$*q_0q_1q_2$	$q_0q_1q_2$	q_0q_3

	0	1
q_0q_3	q_0q_1	$q_0q_3q_4$
$*q_0q_3q_4$	q_0q_1	$q_0q_3q_4$

	0	1
$\rightarrow q_0$	$q_0q_1q_2$	$q_0q_3q_4$
q_1	q_2	—
q_3	—	q_4
$*q_2$	—	—
$*q_4$	—	—
$*q_0q_1q_2$	$q_0q_1q_2$	q_0
$*q_0q_3q_4$	$q_0q_1q_2$	$q_0q_3q_4$



Q:- Convert the following NFA to DFA:-

State	0	1
→p	p, q	p
q	r, s	t
r	p, r	t
*s	—	—
*t	—	—

Solⁿ:-

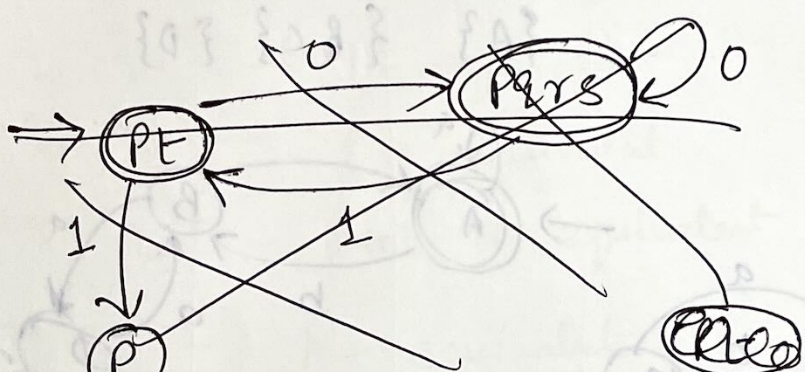
	0	1
→p	p, q	p
q	r, s	t
r	p, r	t
*s	—	—
*t	—	—
p, q	pqrs	pt
*rs	pr	t
pr	pqr	pt
pqr	pqrs	pt
~~pqs~~		
*pqrs	pqrs	pt
*pt	pq	p

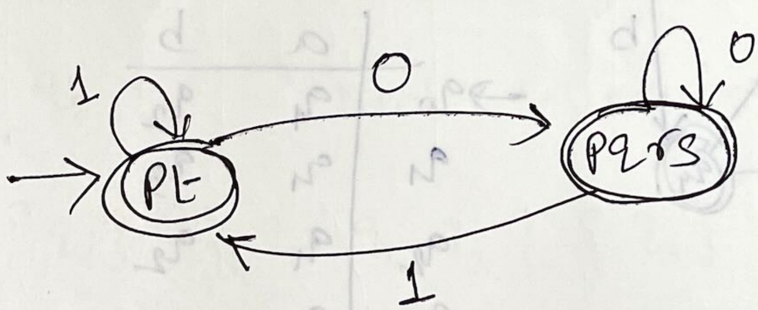
pt

rs

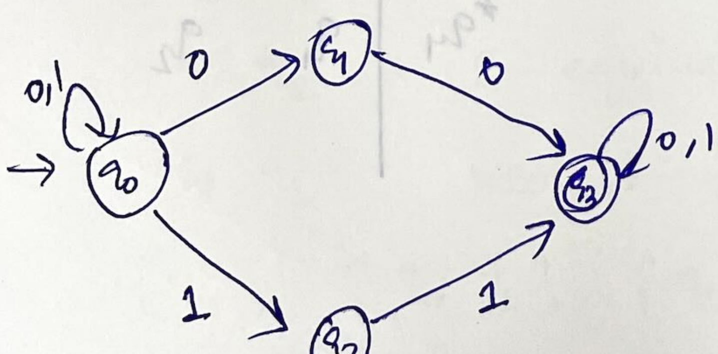
pqrs

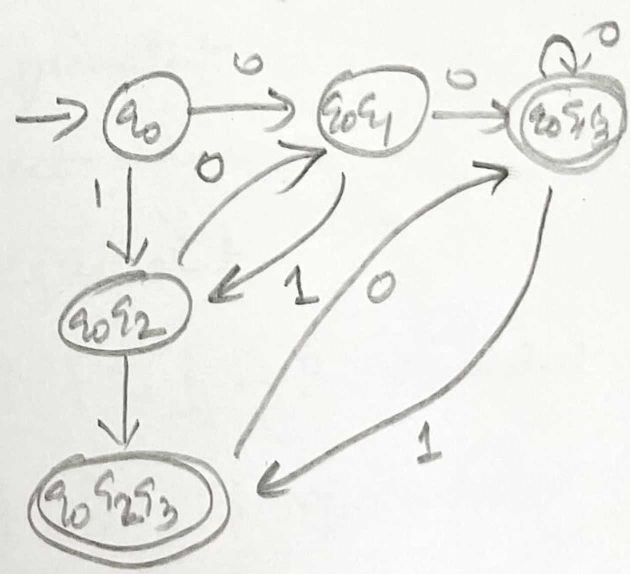
	0	1
→*pt	pqrs	pt
q	rs	t
*rs	pr	t
*s	—	—
*t	—	—
*pqrs	pqrs	pt
pr	pqr	pt



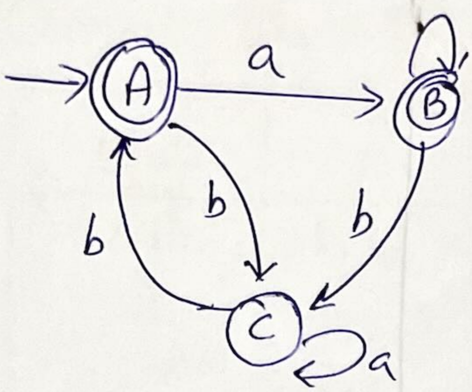


Q!-



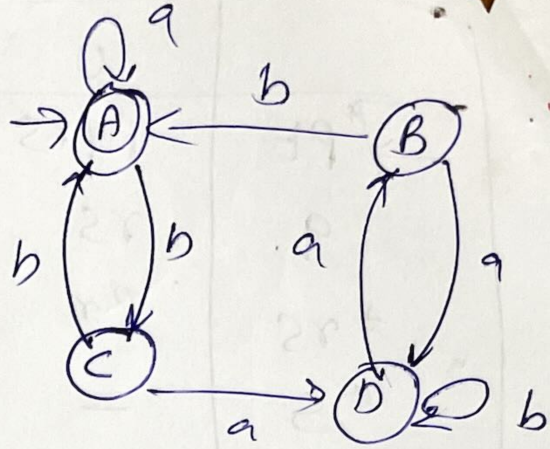


Q:-



{A, B} {C}

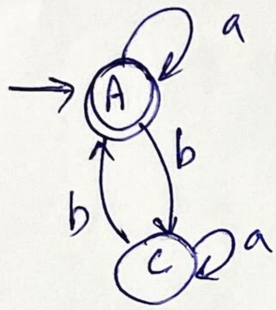
Q:

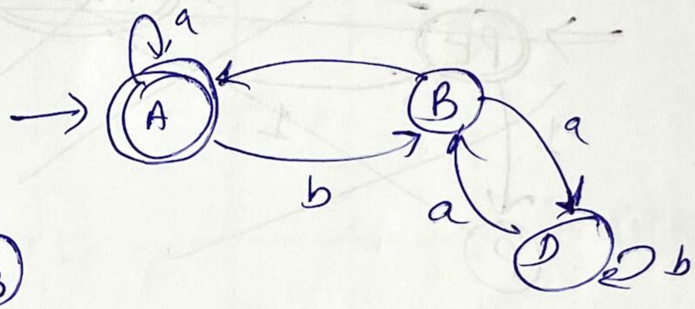


{A} {B, C, D}

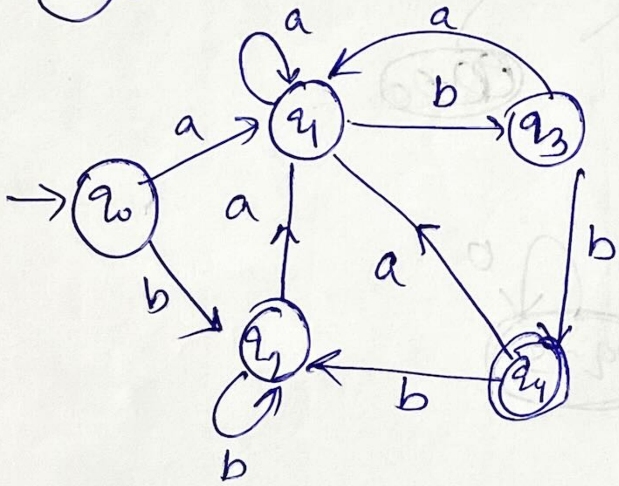
{A} {B, C} {D}

{A} {B, C} {D}

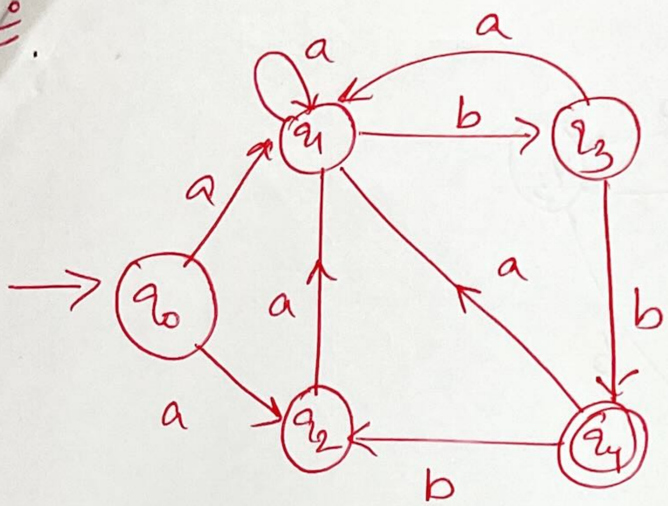




Q:



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



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

Step 1: $\{q_4\}$ $\{q_0\ q_1\ q_2\ q_3\}$

- $q_0\ q_1$ — 1- equivalent
- $q_0\ q_2$ — 1- equivalent
- $q_0\ q_3$ — Not 1- equivalent
- $q_1\ q_2$ — 1- equivalent
- $q_1\ q_3$ — Not 1 -equivalent
- $q_2\ q_3$ Not 1- equivalent

$\{q_4\}$ $\{q_0\ q_1\ q_2\}$ $\{q_3\}$ – 1 - equivalent

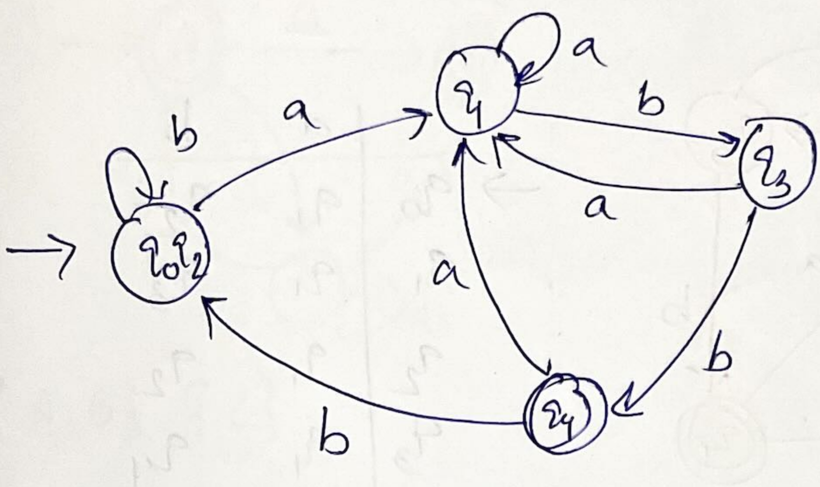
Step 2: $q_0\ q_1$ — Not 2- equivalent

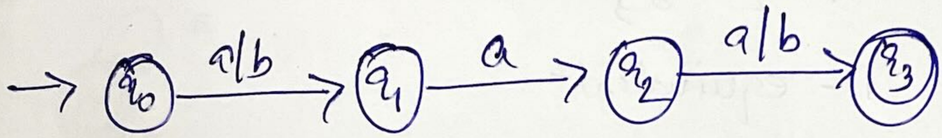
$q_0\ q_2$ - 2 - equivalent

$q_1\ q_2$ - Not 2- equivalent

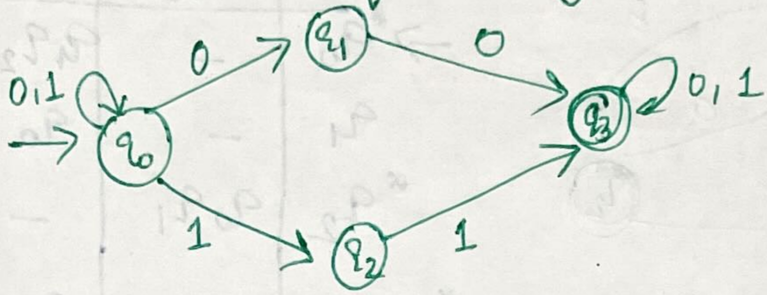
$\{q_4\}$ $\{q_0\ q_2\}$ $\{q_1\}$ $\{q_3\}$ – 2 - equivalent

Step 3: $\{q_4\}$ $\{q_0\ q_2\}$ $\{q_1\}$ $\{q_3\}$ - 3 - equivalent


a
b
a
q1
b
q3
a
q0q2
a
b
q4
b


a/b
a
a/b
q0
q1
q2
q3

Q: Convert the following NFA to DFA.

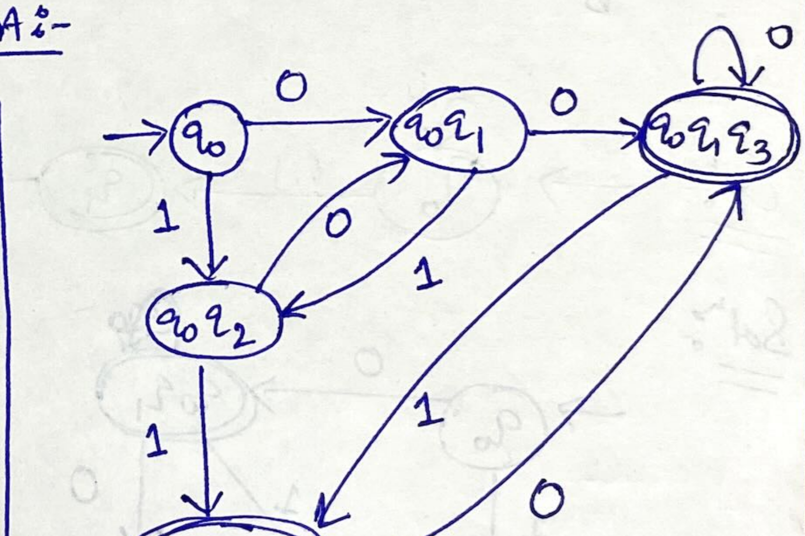


Solⁿ: Transition table for NFA:-

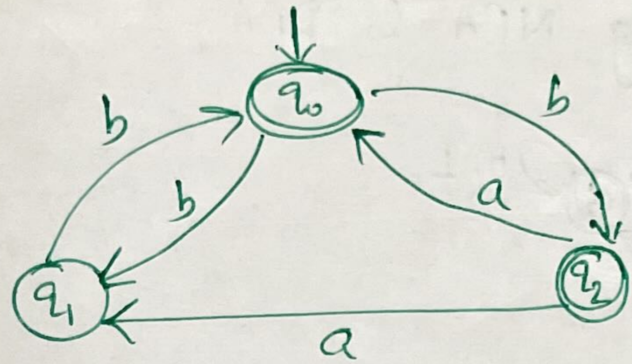
	0	1
$\rightarrow q_0$	q_0q_1	q_0q_2
q_1	q_3	—
q_2	—	q_3
$*q_3$	q_3	q_3

Transition table for DFA:-

	0	1
$\rightarrow q_0$	q_0q_1	q_0q_2
q_1	q_3	—
q_2	—	q_3
$*q_3$	q_3	q_3
q_0q_1	$q_0q_1q_3$	q_0q_2
q_0q_2	q_0q_1	$q_0q_2q_3$
$*q_0q_1q_3$	$q_0q_1q_3$	$q_0q_2q_3$
$*q_0q_2q_3$	$q_0q_1q_3$	$q_0q_2q_3$

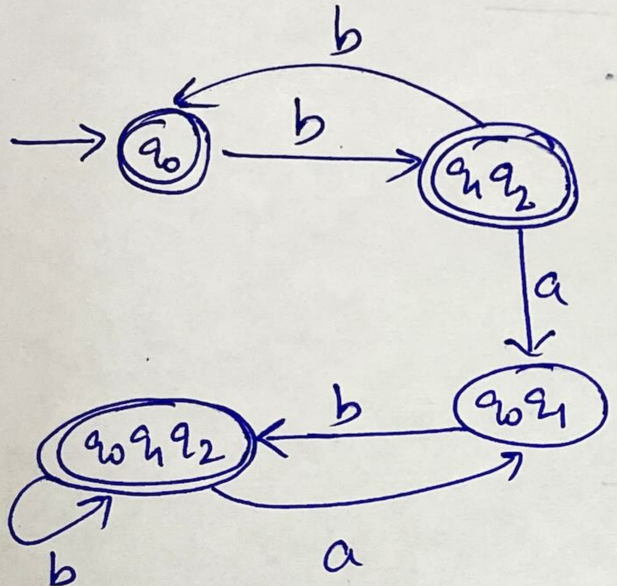


Q:-

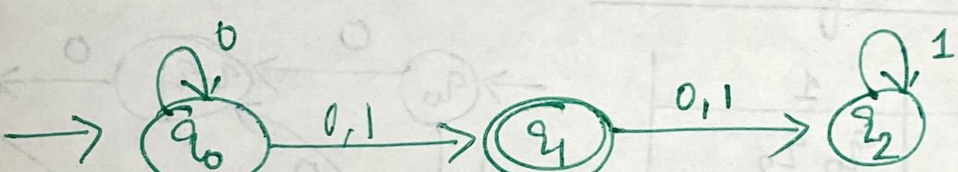


	a	b
$\rightarrow {}^{*}q_0$	–	q_1q_2
q_1	–	q_0
$^{*}q_2$	q_0q_1	–
$^{*}q_1q_2$	q_0q_1	q_0
$^{*}q_0q_1$	–	$q_0q_1q_2$
$^{*}q_0q_1q_2$	q_0q_1	$q_0q_1q_2$

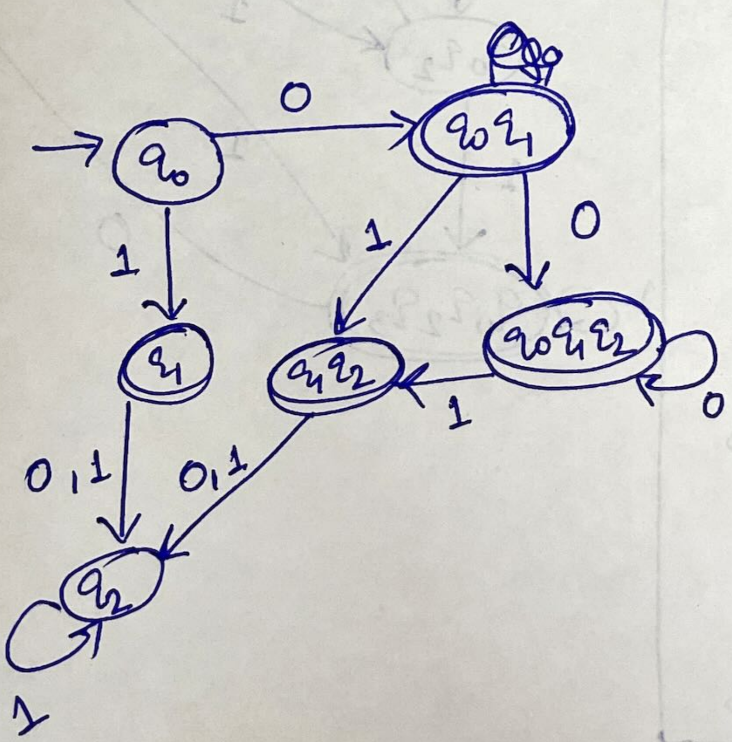
Sol^n:-



Q:-

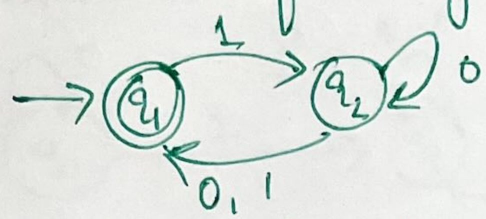


Sol^n:-



	0	1
$\rightarrow q_0$	q_0q_1	q_1
$^{*}q_1$	q_2	q_2
q_2	–	q_2
$^{*}q_0q_1$	$q_0q_1q_2$	q_1q_2
$^{*}q_1q_2$	q_2	q_2
$^{*}q_0q_1q_2$	$q_0q_1q_2$	q_1q_2

Q: Consider the following NFA:-

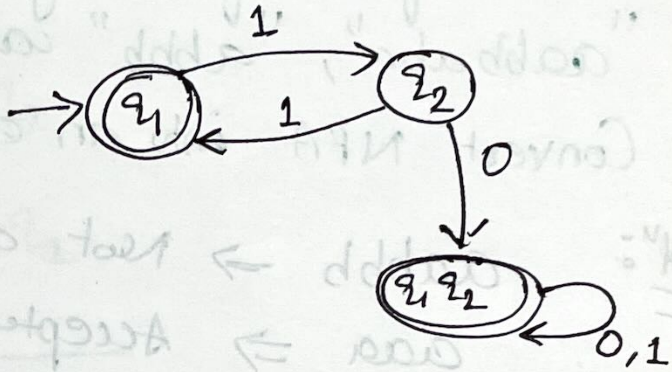


a) Give the state transition of DFA.

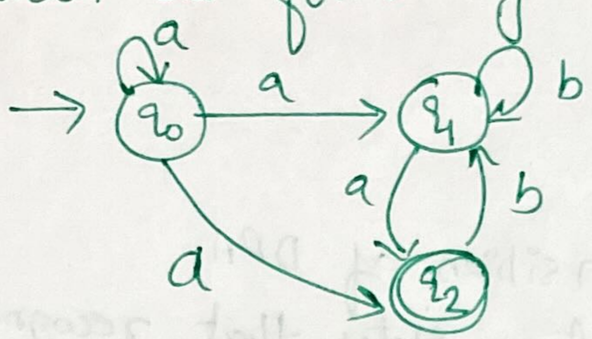
b) Give a NFA with two states that recognizes $\overline{L(N)}$.

Solⁿ:

	0	1
$\rightarrow * q_1$	—	q_2
q_2	$q_1 q_2$	q_1
$* q_1 q_2$	$q_1 q_2$	$q_1 q_2$



Q6: Consider the following NFA:-



a) which of the following strings "aabbb", "aaa", "aba", "aabbaba", "abbb" are accepted by the NFA?

b) Convert NFA into an equivalent DFA.

Solⁿ:

aabbb ⇒ Not accepted

aaa ⇒ Accepted

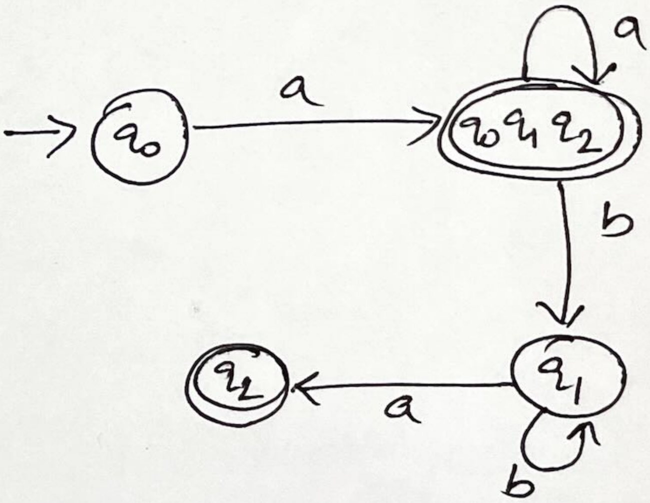
aba ⇒ Accepted

aabbaba ⇒ Accepted

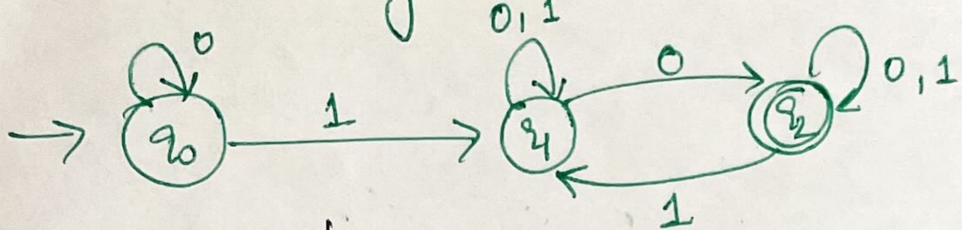
abbb ⇒ Not accepted

	a	b
$\rightarrow q_0$	$q_0q_1q_2$	–
q_1	q_2	q_1
$*q_2$	–	q_1

	a	b
$\rightarrow q_0$	$q_0q_1q_2$	–
q_1	q_2	q_1
$*q_2$	–	q_1
$*q_0q_1q_2$	$q_0q_1q_2$	q_1

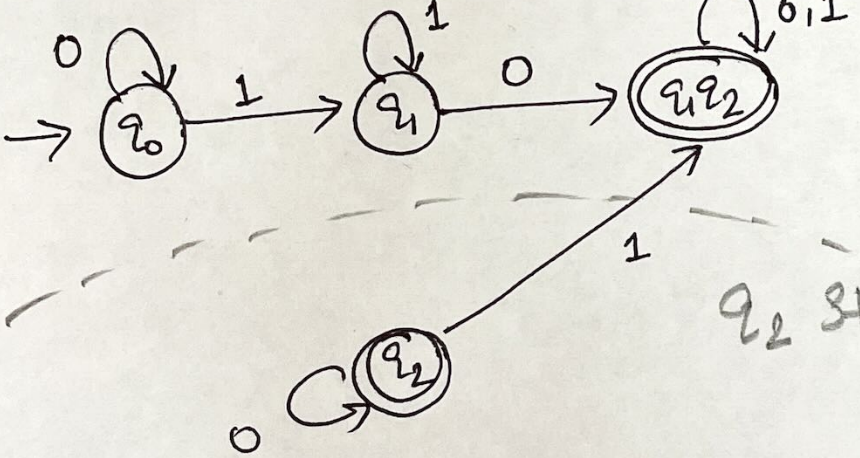


Q: Convert the given NFA to DFA.



Solⁿ:

	0	1
→ q_0	q_0	q_1
q_1	$q_1 q_2$	q_1
* q_2	q_2	$q_1 q_2$
* $q_1 q_2$	$q_1 q_2$	$q_1 q_2$



q_2 State can be omitted.