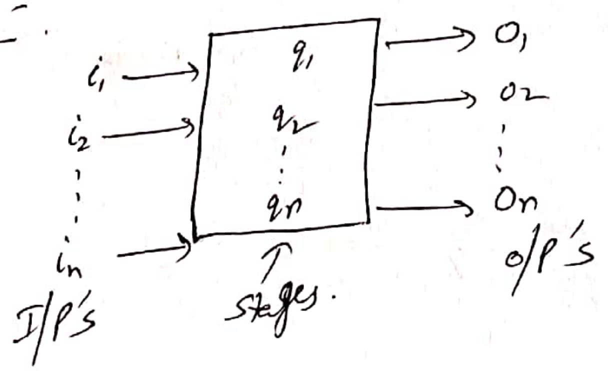
Automata :-



Characteristics :-

① I/P :- At each of the discrete instance of time t_1, t_2 --- & so on and the i/p variable i_1, i_2 -- i_n each of which can take the finite no. of fixed value from the i/p alphabets.

② Output :- O_1, O_2 --- O_n are the o/p of the model.

③ States :- At any instant of time, the automata can be in one of the states q_1, q_2 -- q_n.

④ State Relation :- The next state is determine by present state & present i/p.

⑤ Output Relation :- The o/p is related to i/p & corresponding state.

Scanned with CamScanner

* Finite Automaton :-

$(Q, \Sigma, q_0, \delta, f)$

- Q → State
- Σ → i/p alphabet
- q_0 → initial state
- δ → transition fun.^{c}
- f → final state

$\delta \rightarrow I \times Q \rightarrow O$

(I → i/p, Q → State, O ← i/p)

$\boxed{\text{i.e. } \Sigma \times Q \rightarrow Q}$

$Q \rightarrow$ finite Non-Empty set of state

$\Sigma \rightarrow$ i/p alphabet

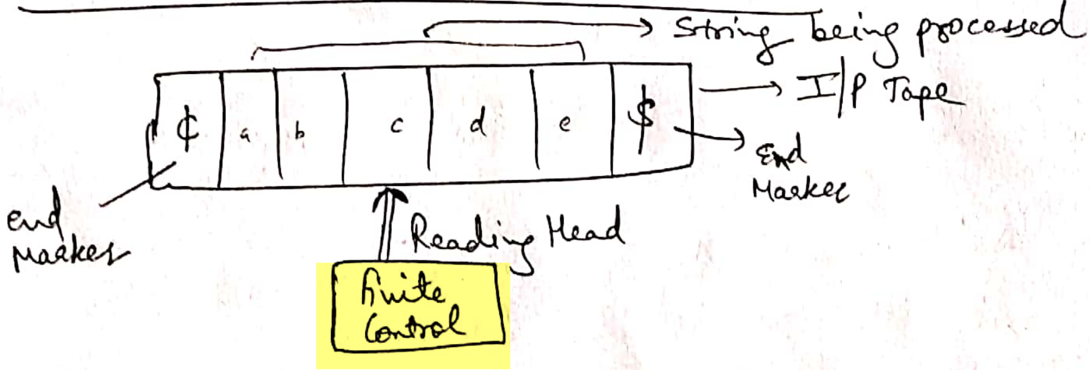
$\delta \rightarrow$ Transition fun.^{c}

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

$f \rightarrow$ final State

$q_0 \rightarrow$ initial State

Block Diagram of Finite Automata :-



Finite Control ⇒ Move only 1 block either left or Right.

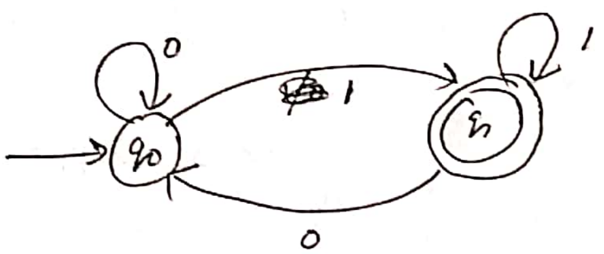
Scanned with CamScanner

1) <u>I/P Tape</u> :- The i/p tape is divided into squares, each square containing the single symbol (Σ). The end of tape contain the end markers. ($¢$, \$).

2) <u>Reading Head</u> :- It examine only one square at a time & can move one square either to the left or to the right.

3) <u>finite Control</u> :- The symbol under read head & the present state of machine.

* Transition <u>System</u> :- / <u>Transition Graph</u> :-



$\rightarrow\bigcirc \Rightarrow$ Initial state

$\circledcirc \rightarrow$ final state

Edges $\rightarrow$ Represent the transition

It is a finite label graph. in which each vertex. represent the state & the directed edges indicate the transition of a state.

The edge are labelled by I/O.

The graph starts at the vertex q_0 & goes along the set of edges & reaches final vertex q_f.

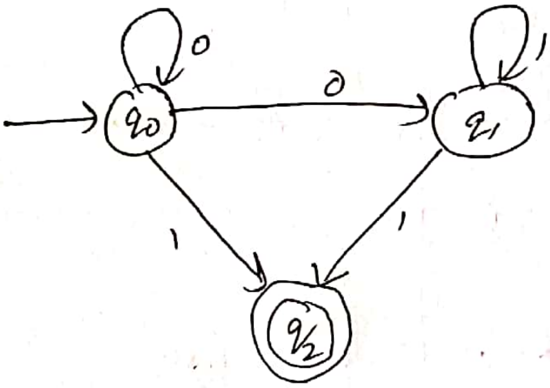
Scanned with CamScanner

2 Types of FA

DFA — NDFA/NFA

* NDFA/NFA :-

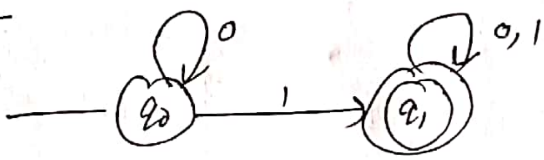
$\boxed{\delta : \Sigma \times Q \to 2^Q}$



	0	1
$\to q_0$	q_0, q_1	q_2
q_1	-	q_1, q_2
q_2	ϕ	ϕ

Some moves of the m/c can't be determine uniquely by the i/p symbol & present state. Such m/c are called NFA.

DFA :-



$\boxed{\delta : \Sigma \times Q \to Q}$

Transition Table :-

States	0	1
$\to q_0$	q_0	q_1
q_1	q_1	q_1

Appl.n of F.A :-

1. Lexical Analyser :- to check validity (organize token)
2. Text editor
3. Spell checker
4. Sequencial circuit Diagram

Equivalence of NFA :-

$$NDFA \xrightarrow{Convert} DFA$$

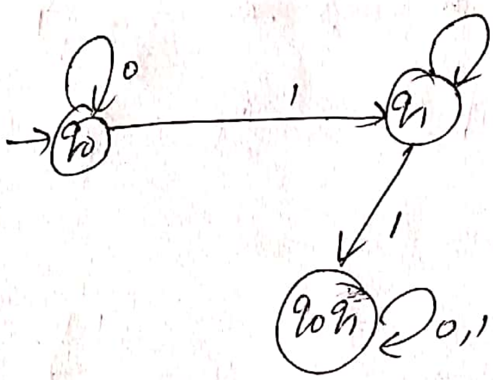
Q) Given :-

State	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_1	$q_0\ q_1$

⇓

DFA :-

State	0	1
$[q_0]$	$[q_0]$	$[q_1]$
$[q_1]$	$[q_1]$	$[q_0\ q_1]$
$[q_0 q_1]$	$[q_0\ q_1]$	$[q_0 q_1]$
ϕ	ϕ	ϕ



Operations on formal language:-

language:

Union:-

$x \in L_1 \cup L_2$

$x \in L_1$ or $x \in L_2$

Eg:- $\{0, 11, 01\} \cup \{1, 01, 110\}$

$\Rightarrow \{0, 1, 01, 11, 110\}$

Intersection:-

$x \in L_1 \cap L_2$

iff $x \in L_1$ and $x \in L_2$

Eg:- $\{0, 11, 01, 01\} \cap \{1, 01, 110\} = \{01\}$

Complement:-

$L = \{x \mid |x| \text{ is even}\}$ where $x \in \Sigma^*$

$L' \Rightarrow \{x \mid |x| \text{ is odd}$

Reversal of language:-

$L \longrightarrow L^R$

$L^R = \{w^R \mid w \in L\}$

Eg:- $L = \{0, 11, 01, 011\}$

$L^R = \{0, 11, 10, 110\}$

Scanned with CamScanner

language Concatination:-

$$L_1 L_2 \Rightarrow \{xy \mid x \in L_1 \text{ \& } y \in L_2\}$$

Eg:- $\{a, ab\}\{b, ba\}$

$\Rightarrow \{ab, aba, abb, abba\}$

Note $\Rightarrow$ (1) $L_1 L_2 \neq L_2 L_1$ in general

(2) $\cancel{L\phi = \phi}$

(3) $\cancel{L(\epsilon) = L = \{\epsilon\}}$

$$L^* = L^0 \cup L^1 \cup L^2 \cdots$$

$$L^+ = L^1 \cup L^2 \cup L^3 \cdots$$

Eg:- $L = \{a, ab\}$

$L^* = L^0 \cup L^1 \cup L^2 \ldots$

$= \{a\} \cup \{a, ab\} \cup (aa, aab, aba, abab) \ldots$

Construction of DFA :-

A) Type-01 Problem :-

In Type-01 problem, we will discuss the Construction of DFA for languages consisting of strings ending with a particular substring.

Step-01 :-
- Determine the minimum no. of states required in DFA
- Draw those states.

Use the following Rule :-

Rule ⇒ Calculate the length of substring.
⇒ All the string ending with 'n' length substring will always require minimum (n+1) states in the DFA

Step-02 :-
- Decide the strings for which DFA will be constructed.

Step-03 :-
- Construct a DFA for the strings decided in step ②

Remember the following Rule while constructing DFA :-

Rule ⇒ while constructing DFA
- Always prefer to use the existing path.
- Create a new path only when there exists no path to go with.

Scanned with CamScanner

Step4:-

(i) Send all the left possible combinations to the starting state.

(ii) Do not send the left possible combinations over the dead state.

Practice Problems :-

(1) Draw a DFA for the language accepting strings ending with '01' over an i/p ~~alphabet~~ alphabet $\Sigma = \{0,1\}$.

Sol.ⁿ ⇒ R.E. ⇒ $(0+1)^* 01$

Step-01:- (i) All ~~string~~ end with '01'

(ii) length of substring ⇒ 02

Thus, Min. no. of states required in the DFA = 2+1 = 3

It suggests that ~~minized~~ minimized DFA will have 3 states.

Step 2 ⇒ We will construct DFA for the following strings.

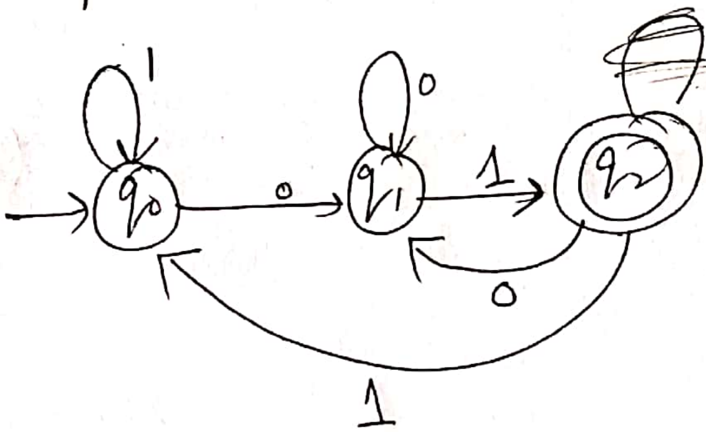
— 01

— 001

— 0101

Scanned with CamScanner

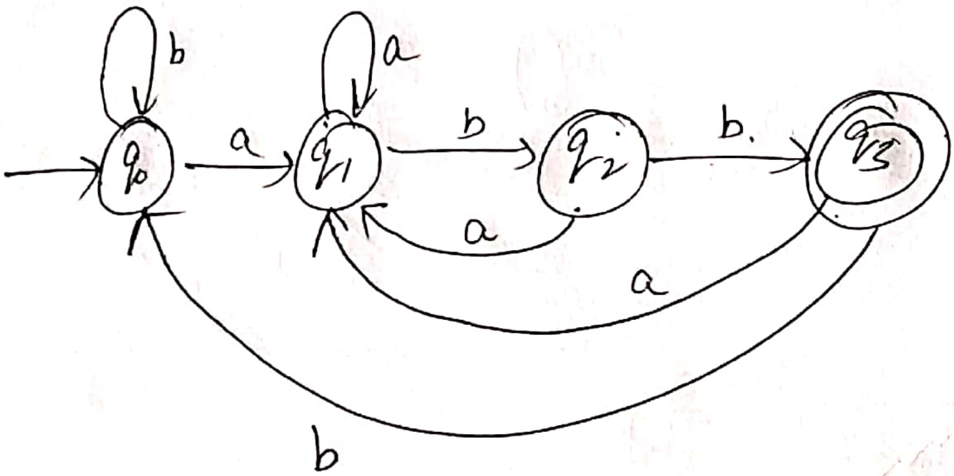
Step 3:- Required DFA is



(2) Draw a DFA for the language accepting strings ending with 'abb' over input alphabets $\Sigma = \{a, b\}$

Sol.n ⇒ R.E. ⇒ $(a+b)^* abb$

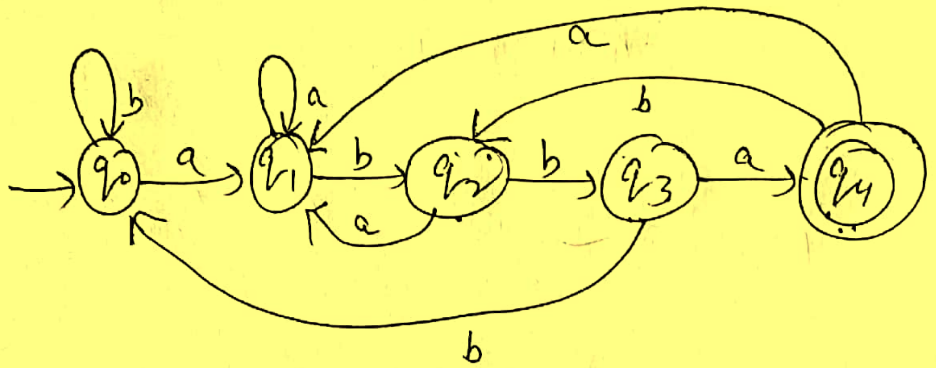
Step 1 ⇒ "abb"
Substring length ⇒ 3
Min. DFA states = 3+1 = (4)



Scanned with CamScanner

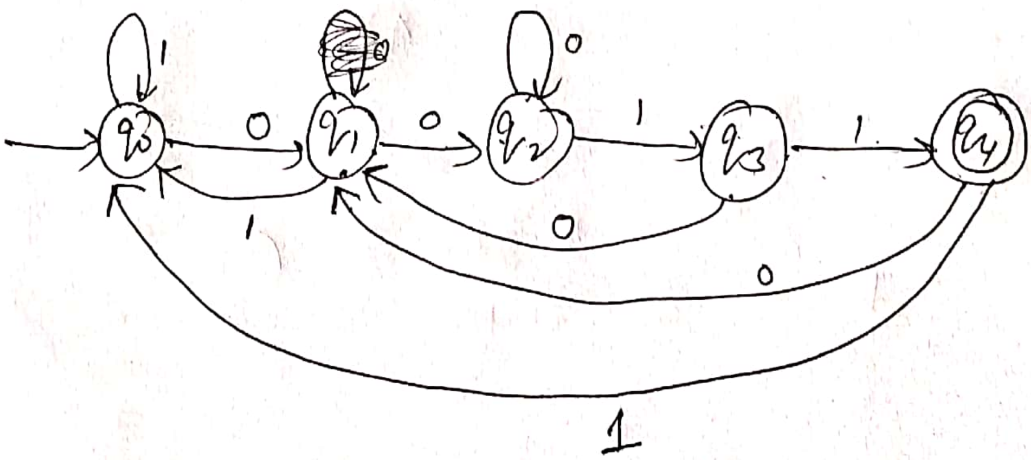
③ Draw DFA for the language accepting string ending with 'abba' over alphabet $\Sigma = \{a, b\}$

Sol.ⁿ→ $4+1 \Rightarrow$ 5 min. possible states



④ Draw DFA for the language accepting strings ending with '0011' over input alphabets $\Sigma = \{0, 1\}$

Sol.ⁿ→



Scanned with CamScanner

B) Type-02 problem :-

In this problem, we will discuss the construction of DFA for languages ~~containing~~ consisting of strings starting with a particular substring.

Steps to construct DFA :-

Step 1 → ⊙ Determine the min. no. of states required in DFA.

⊙ Draw those states.

Use the following rule to determine min. no. of states :-

Rule :- Calculate the length of string.

⇒ All strings starting with 'n' length substring will always require minimum $(n+2)$ states in DFA.

Step 2 :- ⊙ Decide the string for which DFA will be constructed.

Step 3 :- ⊙ Construct the DFA for which string decided in step (2).

Scanned with CamScanner

Rule ⇒ while Constructing a DFA

- Always prefer to use the existing path.
- Create a new path only when there exists no path to go with.

Step 4:-

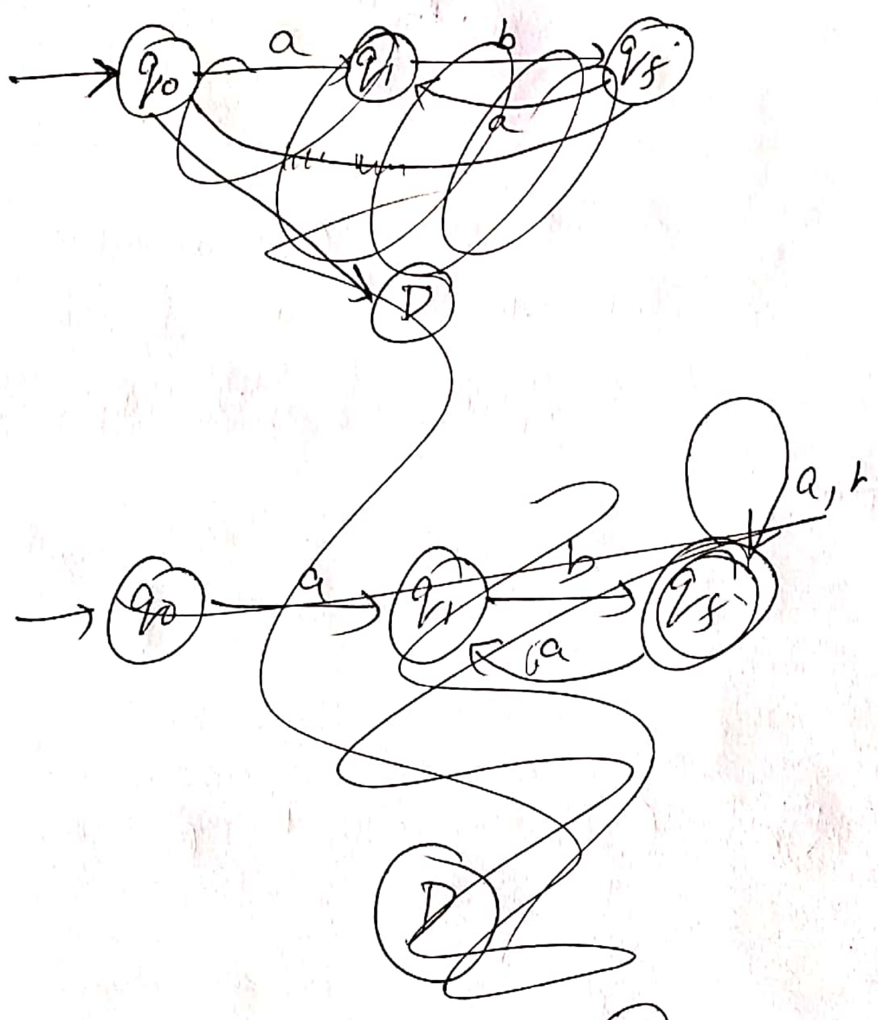
- Send all the left possible combinations to the dead state.
- Do not send the left possible combinations over the starting state.

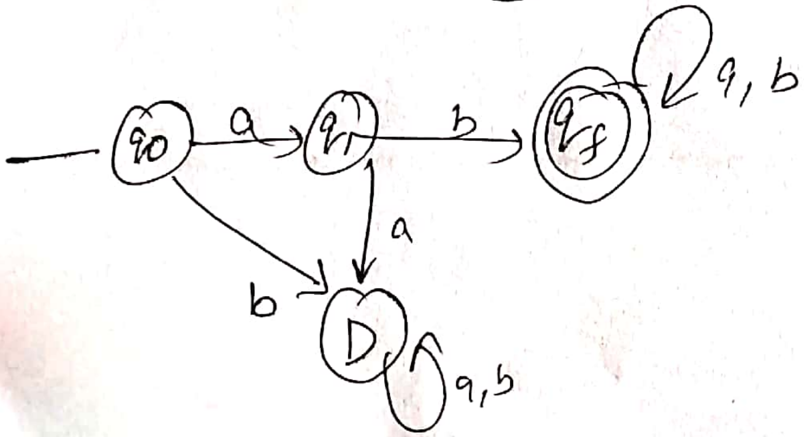
Scanned with CamScanner

Practice Problem:-

① Draw a DFA for the language accepting strings starting with 'ab' over input alphabet $\Sigma = \{a, b\}$

Sol.^{n} $\Rightarrow$ Min. states $\Rightarrow$ $2 + 2 = 4$

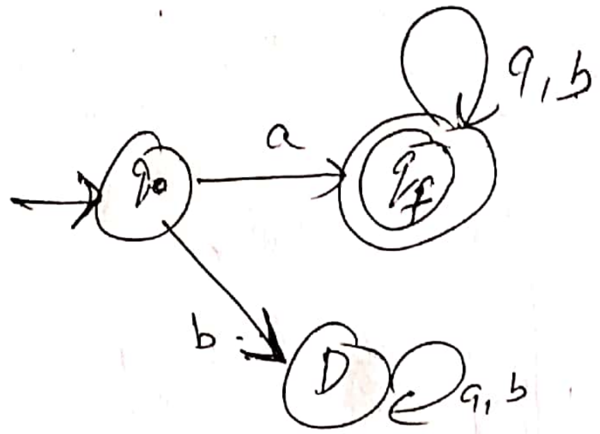




2) Draw a DFA for the language accepting strings starting with a over input alphabet $\Sigma = \{a, b\}$

Sol. ⇒

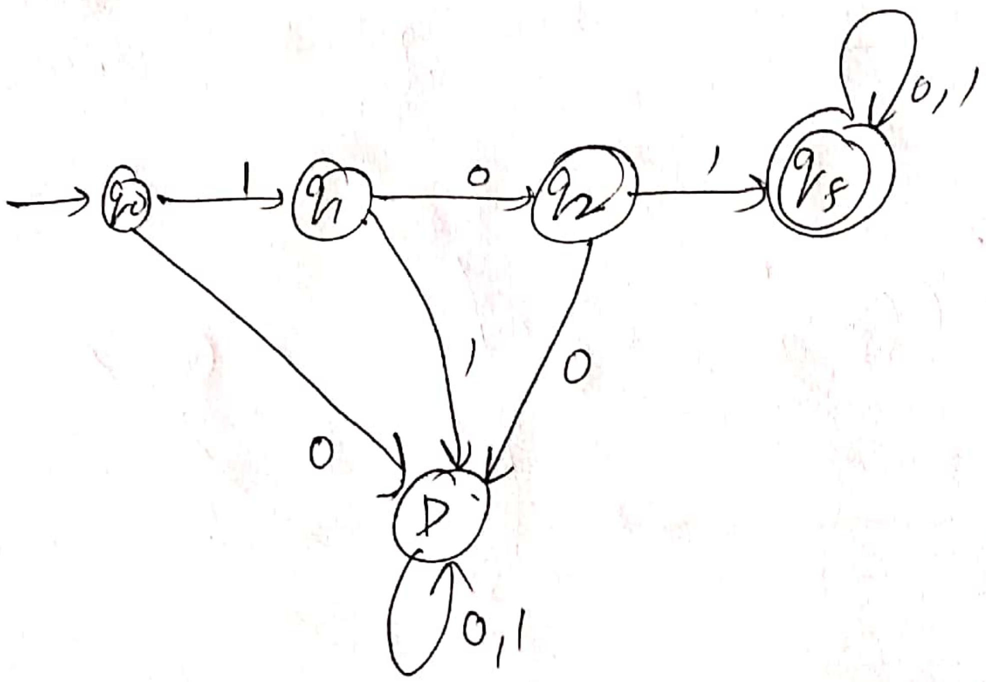
Min. States ⇒ 1+2 = 3



3) Draw a DFA for the language accepting strings starting with '101' over input ~~alpha~~ alphabet $\Sigma = \{0, 1\}$.

Min. States ⇒ 3+2 = 5

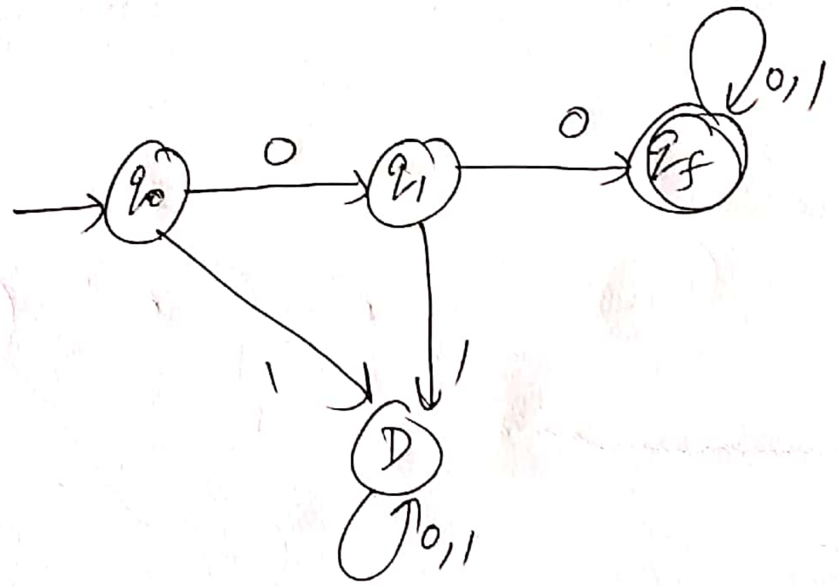
Sol. ⇒



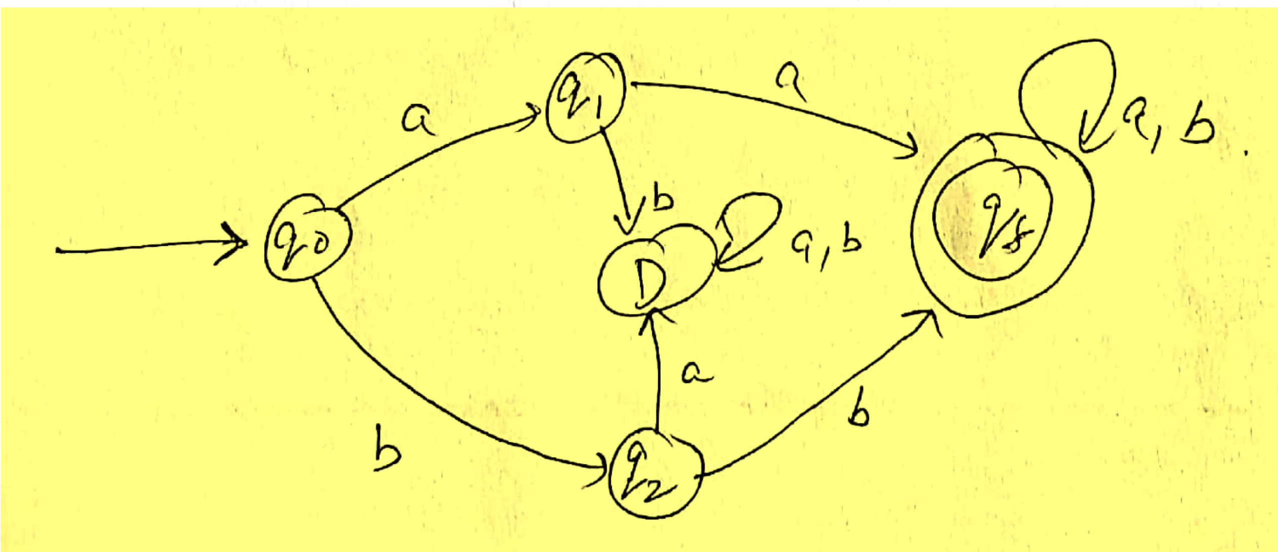
Scanned with CamScanner

(4) Draw a DFA that accepts a language L over input alphabets $\Sigma = \{0, 1\}$ such that L is the set of all strings starting with '00'.

Sol.ⁿ ⇒



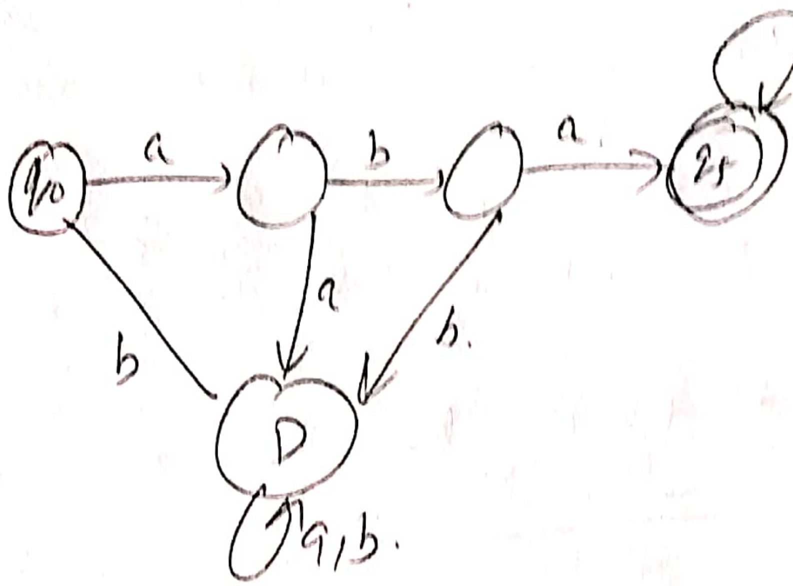
(5) Construct a DFA that accepts a language L over input alphabet $\Sigma = \{a, b\}$ such that L is the set of all strings starting with 'aa' or 'bb'.



Scanned with CamScanner

(6) Construct a DFA that accepts a language L over input alphabet $\Sigma = \{a, b\}$ such that L is the set of all strings starting with '<u>aba</u>'.

Sol.ⁿ



Scanned with CamScanner

Regular Expression :-

Language	R.E
$\{\wedge\}$	$\wedge$
$\{0\}$	0
$\{001\}$	001
$\{0,1\}$	$0+1$
$\{0,10\}$	$0+10$
$\{1,\wedge\}\{001\}$	$(1+\wedge)001$
$\{110\}^*\{0,1\}$	$(110)^*(0+1)$
$\{1\}^*\{1,0\}$	$(1^* 10)$

Scanned with CamScanner

Q ⇒ find R.E for string of even length.

$L = \{00, 01, 10, 11\}^*$

R.E ⇒ $(00 + 01 + 10, 11)^*$

Q ⇒ L = Exactly 3b's Σ a, b

R.E ⇒ $a^* b\, a^* b\, a^* b\, a^*$

Q = L = atleast 3b's over the i/p alphabet

15
10

Σ (a, b)

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

Q ⇒ L = string of length of 6 or less, Σ(0, 1)

for less

$L = \Lambda + 1 + 00 + 01 + 11 + 000 + 001 + \text{---} \; 111111$

R.E ⇒ $(0+1)(0+1)(0+1)(0+1)(0+1)(0+1)$

$= (0 + 1 + \Lambda)^6$ ✓

Scanned with CamScanner

Q→ L = first character 'a' or 'c' followed by any string in 'b'.

R.E → $(a+c)\,b^*$

Q→ L = begin with 'a', $\Sigma(a,b)^*$

R.E → $a(a+b)^*$

Q→ String with odd no. of 1's, $\Sigma(0,1)^*$

R.E ⇒ $0^*10^*(0^*10^*10^*)^*$

odd + Even ⇒ odd

$1(11)^*$

Q→ String ending in 1 & not containing 00.

R.E ⇒ $(1+01)^+$

Scanned with CamScanner