Theory of Computation

(2017)

1. Answer the following questions: [2 marks each]

A) i) write down some applications of Automata.

→ The applications of Automata are as follows:

Finite Automata:
- used in text editors.
- for the implementation of spell checkers.
- For recognizing pattern using RE.

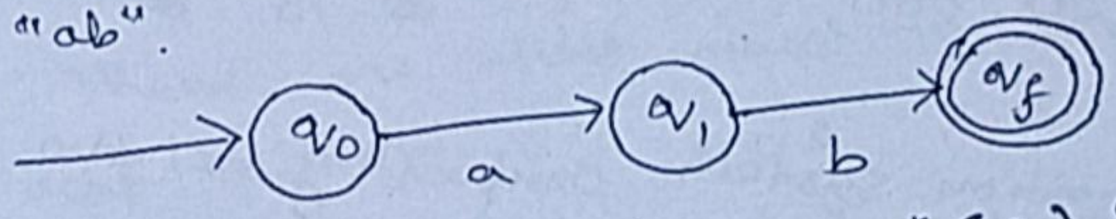
Push Down Automata:
- For solving tower of Hanoi problem.
- For implementation of stack operations.

ii) what is Right Linear Grammar?

→ Right-linear or right regular grammars, in which all non-terminals in right hand side are at the right ends.

iii) Draw the transition diagram for a finite state machine which accepts a String "ab".

→ A finite state machine which accepts an String "ab".



iv) what do you mean by $\epsilon^*(A)$?

→ It means all possible combinations of A with Null.

v) what is dead configuration?

→ Dead configuration means in NFA we did not mention any transition for a particular Symbol and that Symbol comes as input then that String will be silently rejected, but this is not allowed in DFA.

vi) write the names of the operators used in Regular Expressions.

→ Operators of Regular Expressions:

ⓐ Union operator.
ⓑ Concatenation operator.
ⓒ Closure operator.
ⓓ Positive closure operator. or (Kleen closure).

vii) If Regular expression is a^+, what is the language?

→ It is a regular language accepting all possible combination of 'a' without Null.

$L = \{a\}$

$L^+ = \{a, aa, aaa, aaaa, \ldots\ldots\}$.

viii) write the regular expression for a language

$L = \{\epsilon, 0, 00, 000, \ldots\ldots\}$.

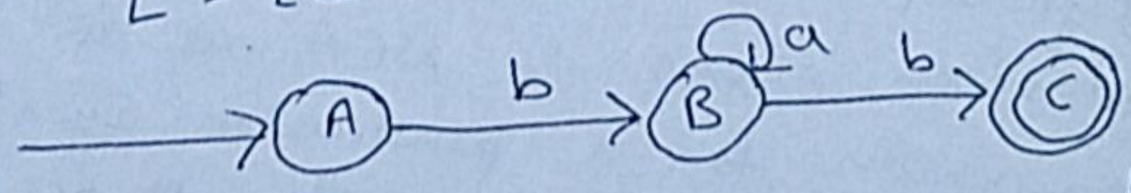
→ The regular expression for a language

$L = \{\epsilon, 0, 00, 000, \ldots\ldots\}$ is is 0^*.

ix) Construct a transition system companding to the regular expression: ba^*b.

→ Regular expression = ba^*b.

$L = \{bb, bab, baab, \ldots\ldots\}$.



x) Explain if the Languages $L = \{a^i b^j \mid i, j \geq 1\}$ is regular or not.

→ $L = \{a^i b^j \mid i, j \geq 1\}$.

This language is not regular as it contradicts the pumping lemma of Regular Language.

xi)

→

xii)

→

xiii)

→

xiv)

(20+
xi)

→

xi) what is ambiguous grammar? (OR what do you mean by ambiguity in grammar?)

→ A grammar of a language is called ambiguous if any of the cases for generating a particular String; more than one left most derivation or more than one right most derivation or more than one parse tree can be generated.

xii) what is unit production?

→ An unit production is a production where the productions of both LHS and RHS are non-terminals

Ex: $A \longrightarrow B$ is an unit production.

[** Unit productions are redundant and should be removed].

xiii) what do you mean by null productio?

→ Null productions are the productions that leads to Null (ϵ).

Ex: $A \longrightarrow \epsilon$.

xiv) when a PDA accepts a String?

→ There are two ways to declare a string accepted by a PDA:

a) Accepted by empty stack.

b) Accepted by final state.

(2018)

xi) what is left linear Grammar?

→ Left linear or left regular grammars in which non-terminals in left hand sides are the left end. It generates the language L. If the finite automata has no final states, then the left linear grammer has an empty set of productions.

ii) Consider the grammar ~~G({S, A}, {a, b}, S, {S → aAb, aA → aaAb, A → ∈})~~.

$G = (\{S, A\}, \{a, b\}, S, \{S \to aAb, aA \to aaAb, A \to \in\})$.

$\longrightarrow$

$G = (\{S, A\}, \{a, b\}, S, \{S \to aAb, aA \to aaAb, A \to \in\})$.

$S \longrightarrow \underline{a}\,Ab$

$\longrightarrow a\,\underline{aAb}\,b$ (by $aA \to aaAb$)

$\longrightarrow aaaAbbb$ (by $aA \to aaAb$)

$\longrightarrow aaabbb$ (by $A \to \in$).

$\longrightarrow a^3 b^3$

$\longrightarrow a^n b^n$

$\therefore L(G) = \{a^n b^n\}$.

iii) Derive the string "aaabbb" from the grammar.

$\longrightarrow$ the above is the answer.

iv) write the names of regular expression operators.

$\longrightarrow$ Operators of Regular Expressions:

ⓐ Union Operator.
ⓑ Concatenation operator.
ⓒ Closure operator.
ⓓ Positive closure operator. or (Kleen Closure).

v) what is positive closure?

$\longrightarrow$ Positive closure or kleen closure can be described as the set of finite length strings that ~~can be~~ allows the use of the same element multiple time i.e. same element can be repeated 1 or n number of times. It is the finite set of all possible strings of all possible length excluding $\in$. It is denoted by Σ^+.

So, $\Sigma^+ = \Sigma^* - \{\in\}$.

vi) If regular expression is $(a+b)^*$, what is language?

→ $(a+b)^*$

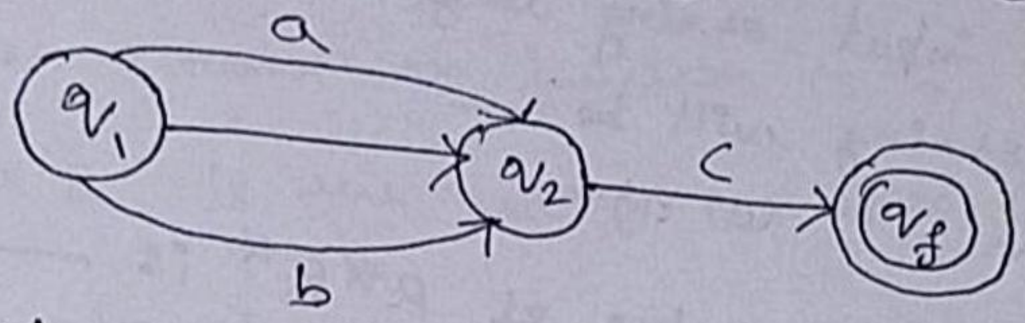
$L = \{a, b, ab, ba, aab, bba, abab, baba, abba \ldots\}$

Set of all strings that contain a, b.

vii) Construct a transition system compounding to the regular expression: $(a+b)c$.

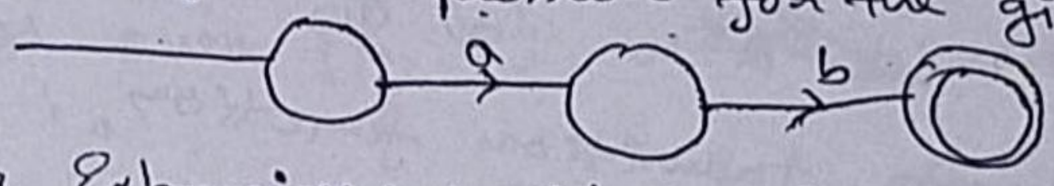
→ $L = \{ac, bc, aac, abc, bbc, bac \ldots\}$

i.e. Set of all strings that ends with c.



viii) write the regular expression for the given finite automata.

→



Regular Expression: $a.(a+b)$

ix) what is unit production?

→ An unit production is a production where the production of both LHS and RHS are non-terminals.

Eg: $A \rightarrow B$ is an unit production.

x) write some applications of automata.

→ The applications of automata are as follows:-

Finite Automata:

- Used in text editors
- for implementation of spell checkers.

Push Down Automata:

- For solving tower of Hanoi problem.
- for implementation of stack operations.

(M.C.Q. was only in 2018)

3.) Choose the correct answers from the given alternatives:

i) Transition function maps:

(a) $\Sigma \times Q \to \Sigma$. (b) $Q \times Q \to \Sigma$

(c) $\Sigma \times \Sigma \to Q$ ~~(d)~~ $Q \times \Sigma \to Q$.

ii) If a NFA contains n states for a problem, the maximum number of state in an equivalent DFA.

(a) $n-1$ ~~(b)~~ $2n$ (c) $3n$ (d) $n!$

iii) In a Moore machine the input string length is n, then the length of output string will be —

~~(a)~~ $n+1$ (b) $n-1$ (c) n (d) $n+n$

iv) If R is a regular expression, value of R^*R^* is —

(a) $2R^*$ (b) R^{*2} ~~(c)~~ R^* (d) ϕ

v) For pushdown Automata, transition function $\delta(a, a, X) = (a, r), X$ is —

~~(a)~~ an input symbol in Σ.

(b) a state in.

(c) are input tape.

(d) a stack symbol.

vi) In PDA, a string is accepted if we end up with —

(a) the final state.

(b) the stack is empty.

(c) the final stack or the stack is empty.

~~(d)~~ The final state and the stack is empty.

vii) Turing machine accepts. —

(a) only type-1 grammar.

(b) only type-2 grammar.

(c) only type-3 grammar.

~~(d)~~ all grammars

2.) a) C
→
A
by
Q:
Σ
δ:

a/c
F

TM
Q:

b) Co

8:

c
→

∴

c) wri
→ -
δ:

1

Broad Questions

2.) a) Give the formal definition of DFA.

→ A deterministic finite automata (DFA) is mathematically represented by its five tuples $(Q, \Sigma, \delta, q_0, F)$ where,

Q: Finite set of states.

Σ: Finite set of alphabet (symbols).

δ: Transition function that takes as argument a state and a symbol and returns a state.

q_0: Initial state.

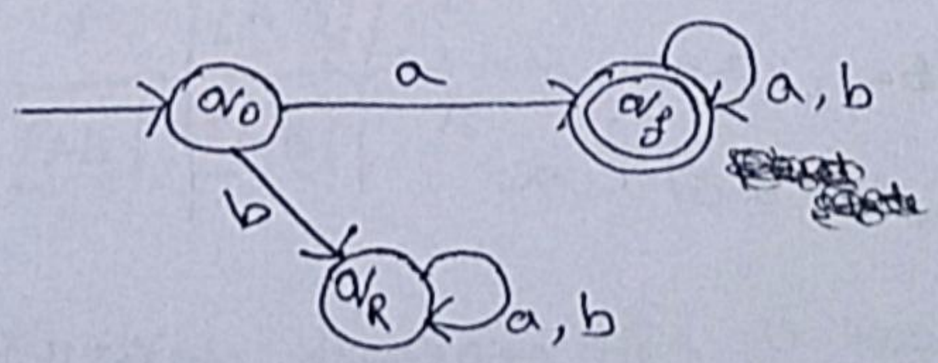
F: Set of final states. We have $q_0 \in Q$ and $F \subseteq Q$.

The transition function is a function in $Q \times \Sigma \rightarrow Q$.

$Q \times \Sigma$ is the set of 2-tuples (q, a) with $q \in Q$ and $a \in \Sigma$.

b) Construct a minimal DFA which accepts set of all strings over $\{a, b\}$, where each string starts with 'a'.

→ $L = \{a, aa, ab, aab, aba \ldots\}$



∴ whenever, 'b' comes as the initial input we will go to the rejected state q_R.

c) write and explain the transition function of NDFA.

→ The transition function of NDFA is represented by δ_N.

δ: Transition function mapping from $Q \times \Sigma$ into 2^Q which is the power set of Q, the set of all subsets of Q.

Here, in case of transition function:

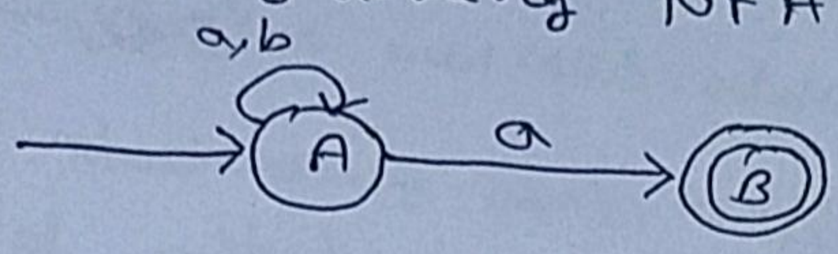
Power set of Q (2^Q) is taken because in case of NDFA, from a state, transition can occur to any combination of Q states. The

difference between the deterministic automata and non-deterministic automata is only in δ. For deterministic automata (DFA), the outcome is a state, i.e. an element of Q; for non-deterministic automata the outcome is a subset of Q.

3.)

a) Convert the following NFA to its equivalent DFA.

→



Transition table for given NFA:

	a	b
A	{A, B}	{A}

$\therefore$ L = { a, ba, bba, aba, aaa, ... }

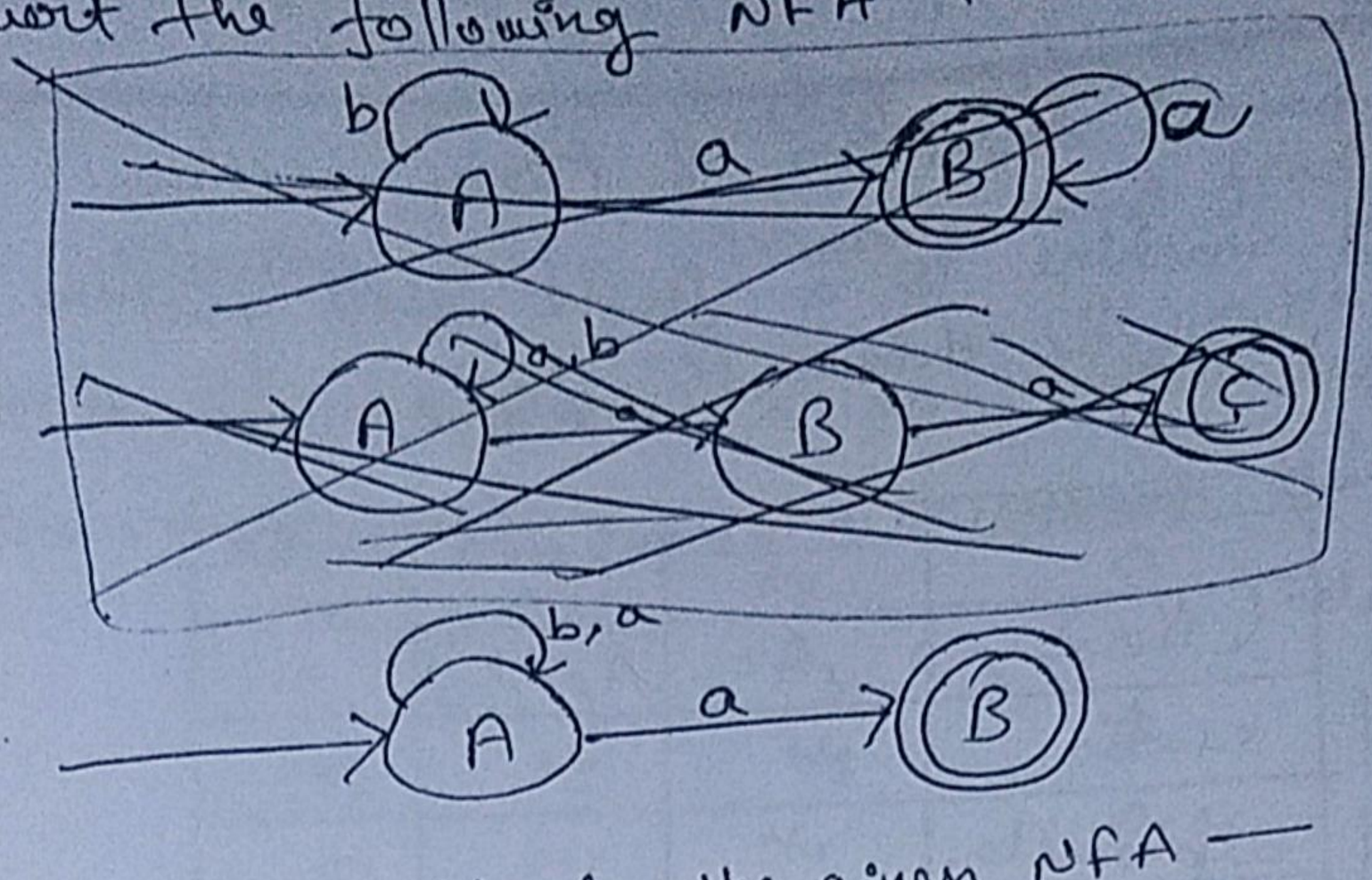
i.e. strings ends with a.

~~**~~ The above question is incorrect and even matrix doesn't have a correct form.

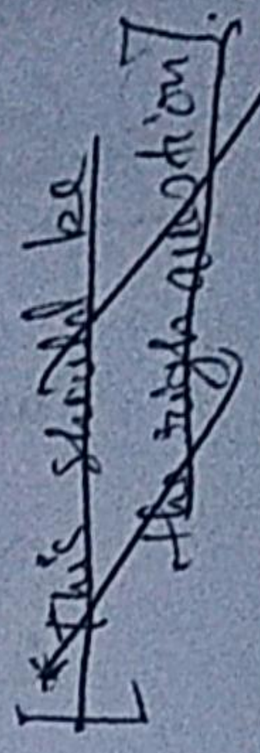
So I am assuming a diagram myself.

3.) a) Convert the following NFA to its equivalent DFA.

→



[∴ given NFA with DFA].



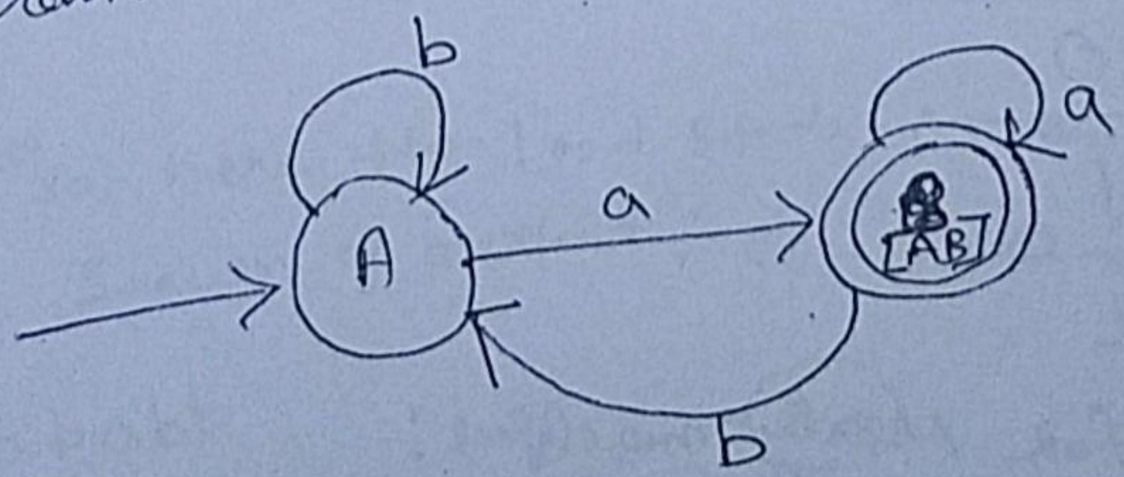
∴ Transition table for the given NFA —

State	a	b
A	{A, B}	A
B	ϕ	ϕ

Now, ~~DFA~~ transition table for its equivalent DFA ——

State	a	b
A	[AB]	A
[AB]	[AB]	A

∴ Transition diagram for its equivalent DFA ——



3.) Give the formal definition of Mealy machine.

→ Mealy machine is a finite state machine where output depends on a machine's - current state and current transition or input. It is represented by six tuples $(Q, \Sigma, \delta, q_0, \Delta, \lambda)$:

Q: Set of states.

Σ: Set of alphabets.

δ: Transition function which is represented by $Q \times \Sigma \rightarrow Q$.

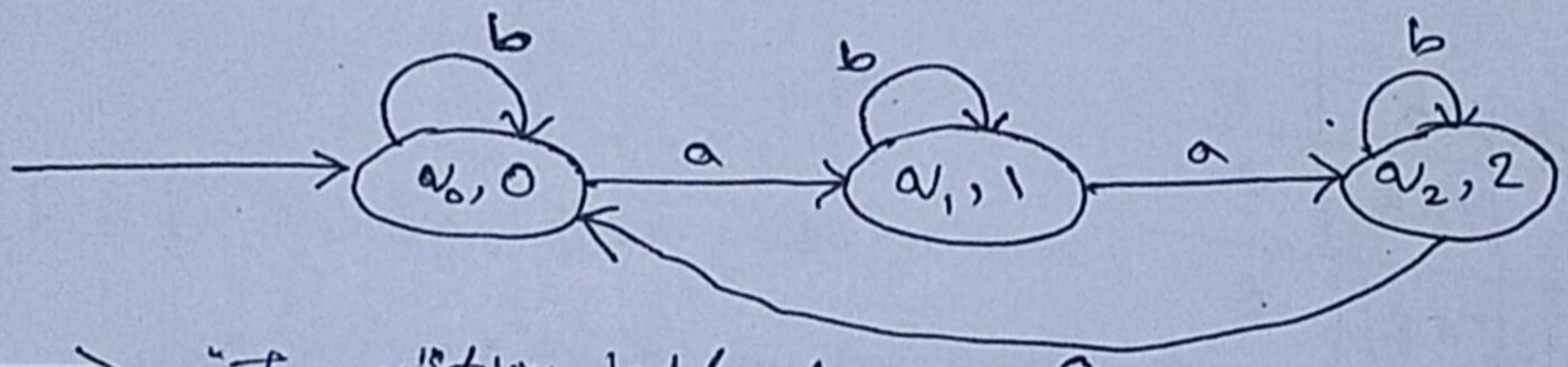
q_0: Initial State.

Δ: Output alphabet set.

λ: Output function which is represented by $Q \times \Sigma \rightarrow \Delta$.

4.)

a) Convert the following Moore machine to its equivalent Mealy machine:



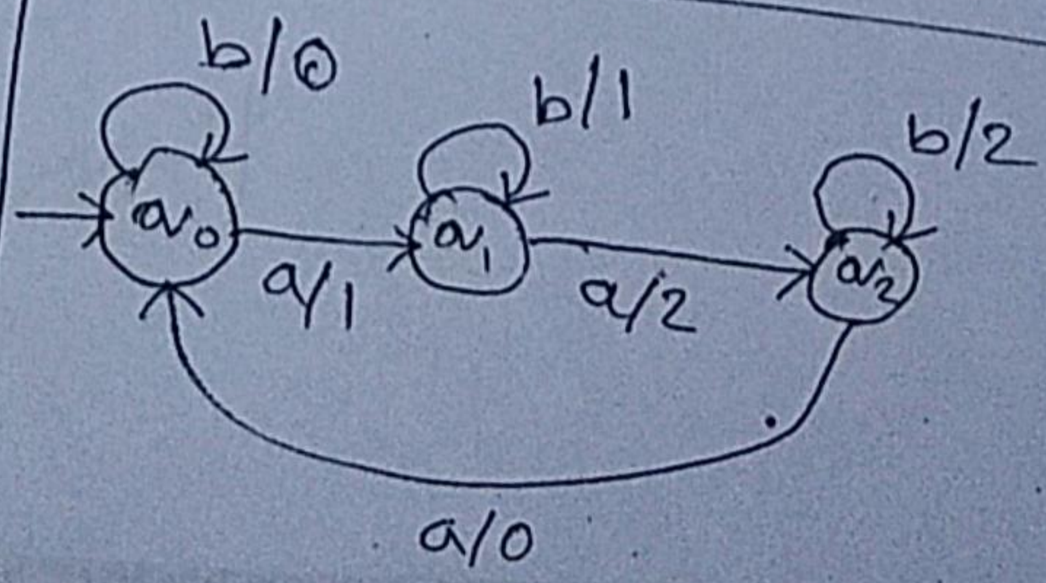
→ Transition table for Moore machine:

Q / Σ	a	b	output
q_0	q_1	q_0	0.
q_1	q_2	q_1	1
q_2	q_0	q_2	2

Now, transition table for Mealy machine:

Q / Σ	a,0/1	b,0/0
q_0	q_1, 1	q_0, 0
q_1	q_2, 2	q_1, 1
q_2	q_0, 0	q_2, 2

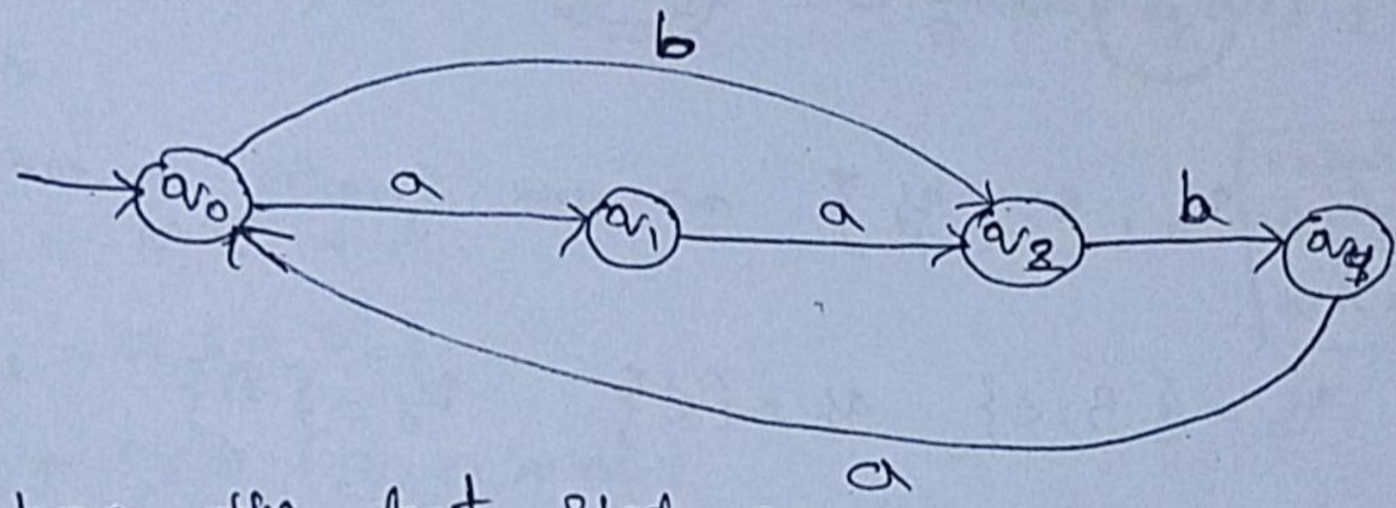
equivalent mealy machine:



the following Moore machine, the input alphabet $\Sigma = \{a, b\}$ and the output alphabet $\Delta = \{0, 1\}$. Run the following input sequences and find the respective outputs. (i) aababab (ii) abbb (iii) ababb.

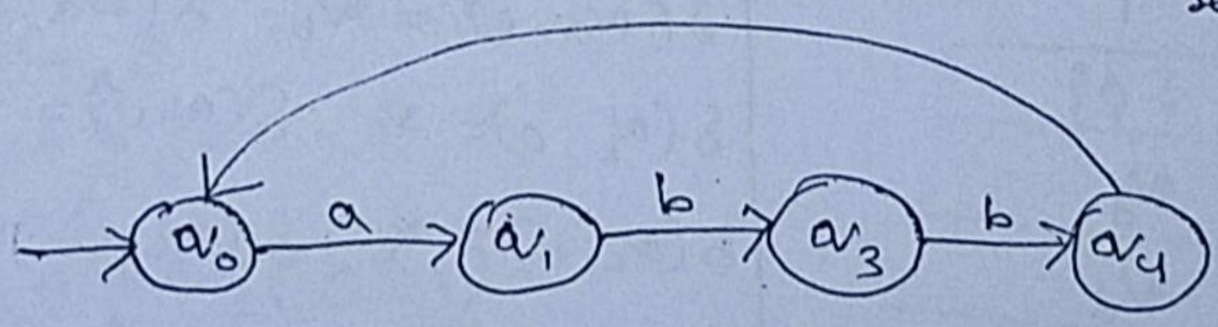
States	a	b	Output
q_0	q_1	q_2	0
q_1	q_2	q_3	0
q_2	q_3	q_4	1
q_3	q_4	q_4	0
q_4	q_0	q_0	0

→ (i) aababab



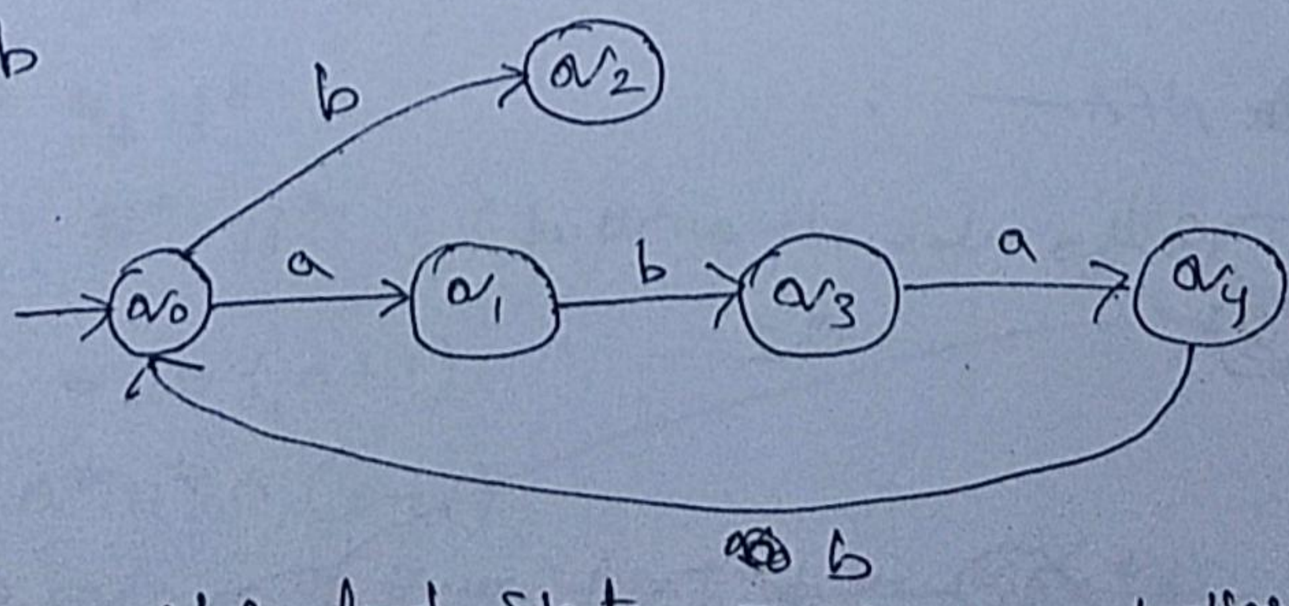
Sol: So, here the last state is q_2, giving the output as '1'.
Sequence is: 0 1 0 0 1.

(ii) abbb



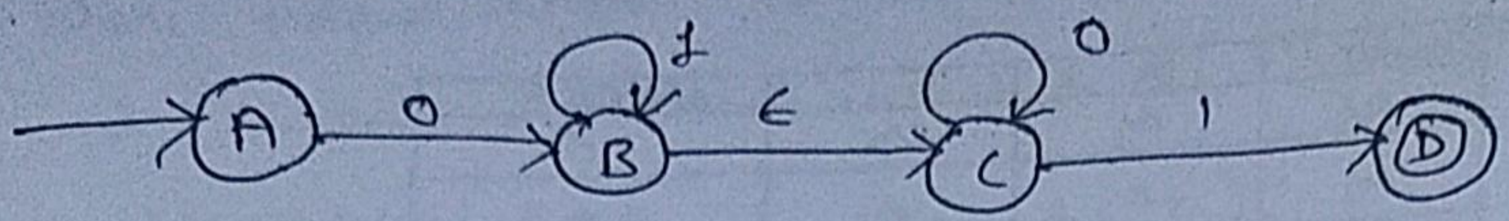
Sol: So, here the last state is q_0, giving the output as '0'
Sequence of output: 0 0 0 0

(iii) ababb



Sol: So, here the last state is q_2, giving the output as '1'.
Sequence of output: 0 0 0 0 1.

5.) a) Convert the following $\epsilon \rightarrow$ NFA to its equivalent NFA.



$\longrightarrow$

Set of new states —

Q': { ϵ-closure(A), ϵ-closure(B), ϵ-closure(C), ϵ-closure(D) }.

= { {A}, {B, C}, {C}, {D} }.

Let, Q': { q_a, q_b, q_c, q_d } as new States in NFA.

where —

q_a = {A} q_b = {B, C} q_c = {C} q_d = {D}

Transition table for NFA —

Σ / Q	0	1
q_a	q_b	{ϕ}
q_b	q_c	q_d
q_c	q_c	q_d
q_d	{ϕ}	{ϕ}
{ϕ}	{ϕ}	{ϕ}

∴ Transition function:

$\delta(q_a, 0) = q_b$, $\delta(q_a, 1) = \{\phi\}$.

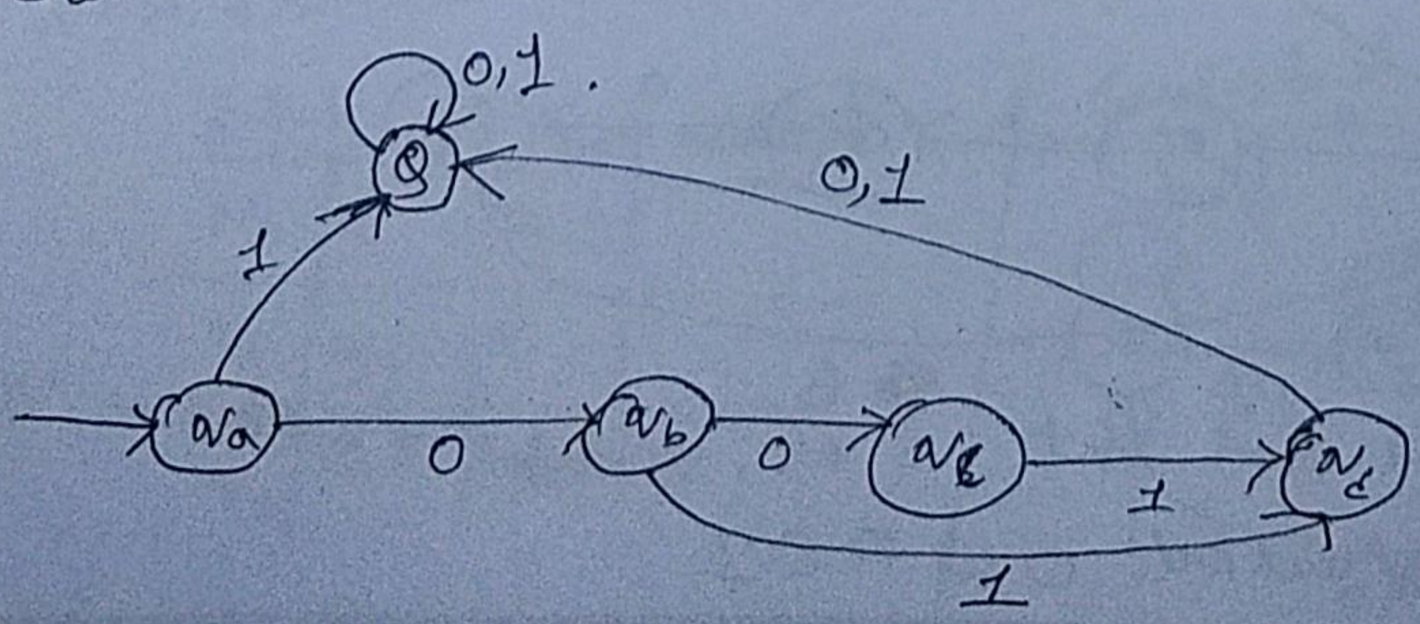
$\delta(q_b, 0) = q_c$, $\delta(q_b, 1) = q_d$.

$\delta(q_c, 0) = q_c$, $\delta(q_c, 1) = q_d$.

$\delta(q_d, 0) = \{\phi\}$, $\delta(q_d, 1) = \{\phi\}$.

$\delta(\{\phi\}, 0) = \{\phi\}$, $\delta(\{\phi\}, 1) = \{\phi\}$.

∴ Equivalent NFA —



b) Gi

set

$\longrightarrow$

Σ =

leng

∴ Regul

ϵ

6.) a) Con

and

$\longrightarrow$

So,

q_1 =

q_2

q_3 =

Applyi

q_1 = 0

$\Rightarrow q_1 = q$

q_3 = (

=

∴ q_2

∴ q_3.

=

Now, th

∴ The

A.

→ Given, $\Sigma = \{a, b\}$. Find out the regular expression for the set of all strings whose length is at most 2.

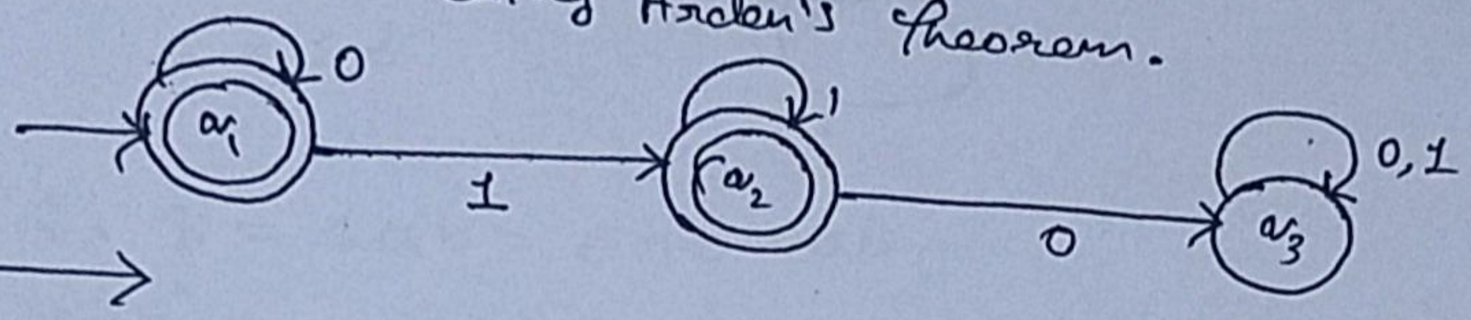
$\Sigma = \{a, b\}$, Regular expression for set of all strings whose length is at most 2.

So, $L = \{\epsilon, a, b, ab, aa, ba, bb\}$

∴ Regular expression:

$\epsilon + a + b(a+b)(a+b)$

6.) a) Construct a regular expression from the given automata using Arden's theorem.



Arden's theorem States:
$R = Q + RP$
$\Rightarrow R = QP^*$

So,

$q_1 = q_1 0 + \epsilon$

$q_2 = q_1 1 + q_2 1$

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

Applying Arden's theorem —

$q_1 = q_1 0 + \epsilon$.

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

$q_3 = q_2 0 + q_3 (0+1)$

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

$\therefore q_2 = q_2 1 + q_2 1$

$= q_1 11^*$

$= 0^* 11^*$ (putting the value of q_1)

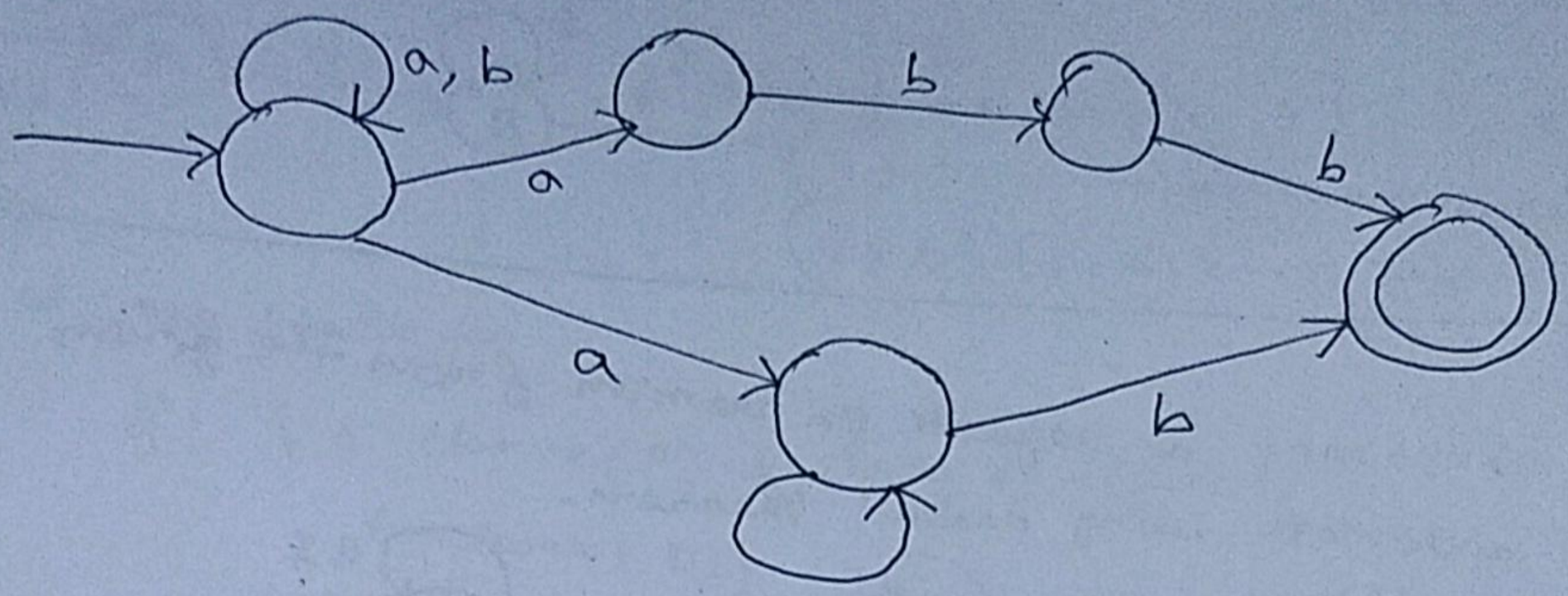
$\therefore q_3 = q_2 0 (0+1)$

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

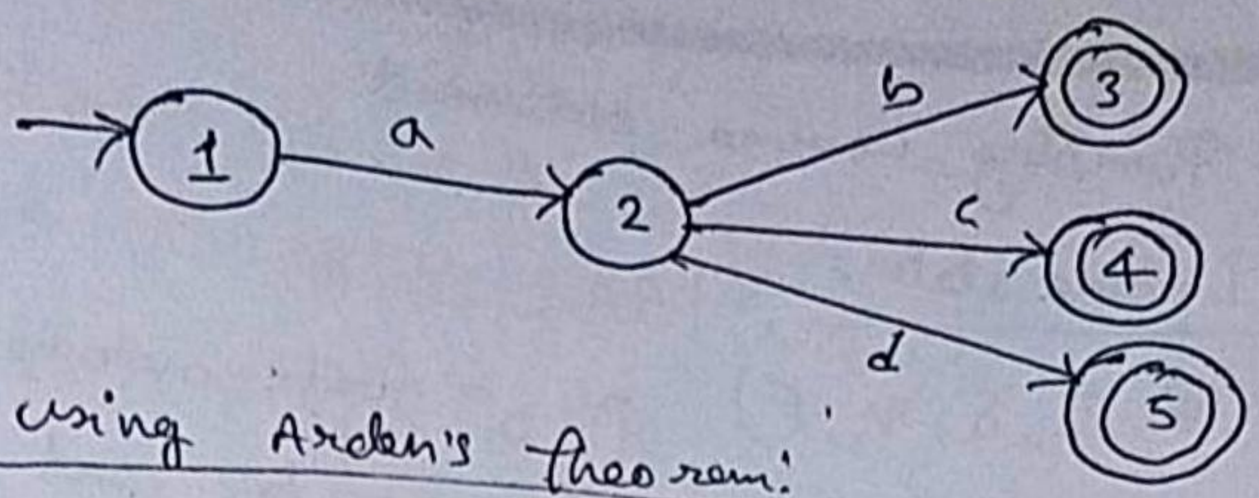
Now, there are only two final States q_1 & q_2;

∴ The regular expression is $(0^* + 0^* 11^*)$.

b) Construct a transition system compounding to the regular expression: $(a|b)^*(abb|a^+b)$.



a) Get the regular expression for the given finite automata:



using Arden's theorem:

$1 = \epsilon$

$2 = 1a$

$3 = 2b$

$4 = 2c$

$5 = 2d$

$\therefore 3 = 1ab = \epsilon ab = ab.$

$\therefore 4 = 1ac = \epsilon ac = ac$

$\therefore 5 = 1ad = \epsilon ad = ad.$

As 3, 4, 5 are the only final state, the regular expression will be ——

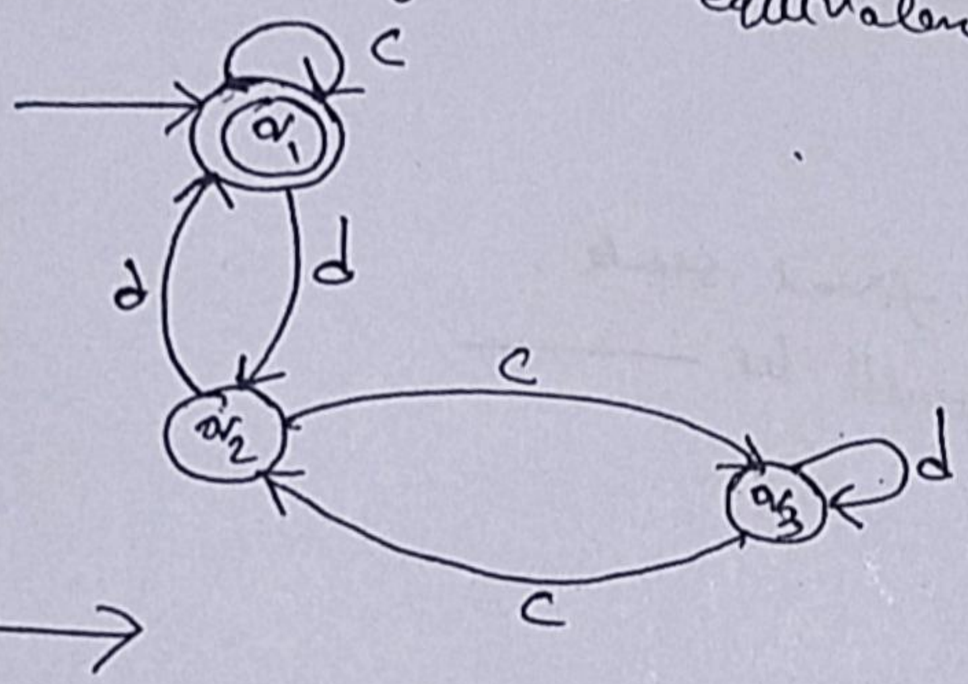
$(ab + ac + ad)$.

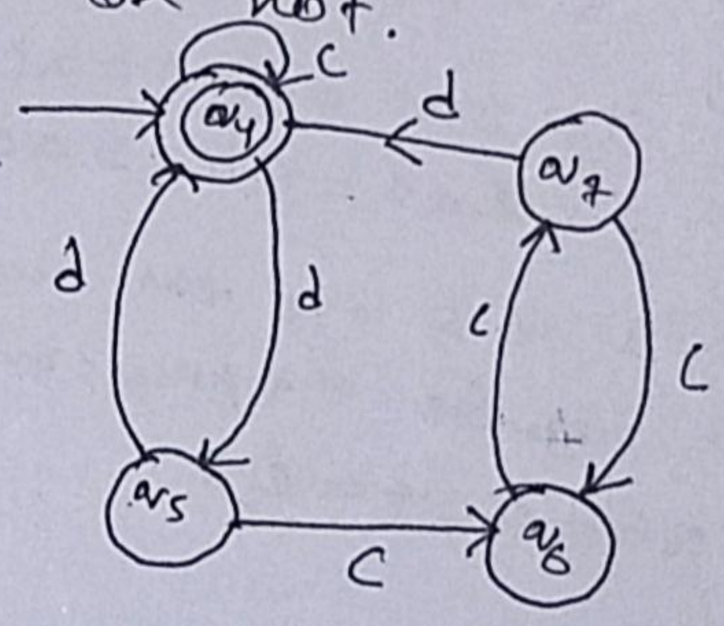
b) Write the Pumping Lemma statement.

→ Pumping Lemma states:

Let, $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automata with n states. Let L be the regular set accepted by M. Let $w \in L$ and $|w| \geq m$, then there exists x, y, z such that $w = xyz$, $y \neq \epsilon$ and $xy^i z \in L$ for each $i \geq 0$.

8) a) Consider the two automata below and determine whether they are equivalent or not.





→

Now, the first condition is satisfied that is both the automata have initial state as final state.

Checking for their equivalence through transition table:

Pair of states	c	d
q_1, q_4	$\frac{q_1}{F}, \frac{q_4}{F}$	$\frac{q_2}{NF}, \frac{q_5}{NF}$
q_2, q_5	$\frac{q_3}{NF}, \frac{q_6}{NF}$	$\frac{q_1}{F}, \frac{q_4}{F}$
q_3, q_6	$\frac{q_2}{NF}, \frac{q_7}{NF}$	$\frac{q_3}{NF}, \frac{q_6}{NF}$
q_2, q_7	$\frac{q_3}{NF}, \frac{q_6}{NF}$	$\frac{q_1}{F}, \frac{q_4}{F}$

As each and every pair of states produces proper combination of final (F) states and non-final (NF) states.

So, the given two Automatas are equivalent.

b) verify weather the grammar $S \to 0B \mid 1A$, $A \to 0 \mid 0S \mid 1AA \mid \wedge$, $B \to 1 \mid 1S \mid 0BB$ generates the string 00110101.

$\longrightarrow$

$S \to 0B \mid 1A$,

$A \to 0S \mid 0S \mid 1AA \mid \wedge$,

$B \to 1 \mid 1S \mid 0BB$.

$\Rightarrow S \to 0\underline{B} \to 00B\underline{B} \to 00B1\underline{S} \to 00B10\underline{B}$

$\to 00B101\underline{S} \to 00B1010\underline{B} \to 00\underline{B}10101.$

$\to 00110101.$

The grammar generates the required string.

9.)

a) Find a reduced grammar equivalent to the grammar G, having production rules P: $S \to AC \mid B$, $A \to a$, $C \to C \mid BC$, $E \to Aa \mid e$.

$\longrightarrow$

Phase 1 —

$T = \{a, C, e\}$

$W_1 = \{A, C, E\}$ from rules $A \to a$, $C \to C$ and $E \to aA$.

$W_2 = \{A, C, E\} \cup \{S\}$ from rule $S \to AC$.

$W_3 = \{A, C, E, S\} \cup \emptyset$.

Since $W_2 = W_3$, we can derive G' as —

$G' = \{\{A, C, E, S\}, \{a, c, e\}, P, \{S\}\}$

where P: $S \to AC$, $A \to a$, $C \to c$.

Phase 2 —

$Y_1 = \{S\}$

$Y_2 = \{S, A, C\}$ from rule $S \to AC$.

$Y_3 = \{S, A, C, a, c\}$ from rules $A \to a$ and $C \to c$.

$Y_4 = \{S, A, C, a, c\}$

Since, $Y_3 = Y_4$. we can derive G'' as —

$G'' = \{\{A, C, S\}, \{a, c\}, P, \{S\}\}$

where P: $S \to AC$, $A \to a$, $C \to c$.

b). Remove unit production for the grammar having production rules, $S \longrightarrow a/Xb/aYa/b/aa$, $X \longrightarrow Y$, $Y \longrightarrow b/X$

$\longrightarrow S \longrightarrow a/Xb/aYa/b/aa$

$X \longrightarrow Y$ } unit production.
$Y \longrightarrow b/X$ }

$\Rightarrow S \longrightarrow a/b/aa/bb/aba$.

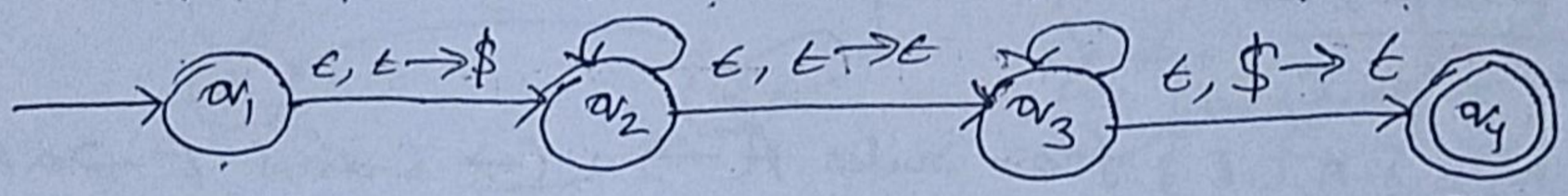
Both the unit production $X \longrightarrow Y$, $Y \longrightarrow X$ be removed

---

10.)

a) Construct a PDA that accepts even palindromes of the form: $L = \{ww^R / w \in (a+b)^+\}$;

$\longrightarrow$



PDA for $L = \{ww^R / w = (a+b)^+\}$.

Initially we put a special symbol '$' into the empty stack. At state q_2, the w is being read. In state q_3, each 0 or 1 is popped when it matches the input. If any other input is given, the PDA will go to a dead state. When we reach the special symbol '$', we go to the accepting state q_4.

b) write a short note on Turing Machine.

→ Turing machine is the most powerful model proposed by Alan Turing in 1936. It is more accurate model for general computer. Turing machine is similar to finite automata having unlimited memory.

A turing machine is represented by 7-tuples.

$M = (Q, \Gamma, b, \Sigma, \delta, q_0, F)$ where:

Q: finite set of states.

Γ : finite set of tape alphabet / Symbols.

b : the blank symbol that is allowed to occur on the tape infinitely often at any step during the computation.

Σ : is the set of input symbols.

δ: A transition function denoted by ——.

$$Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}.$$

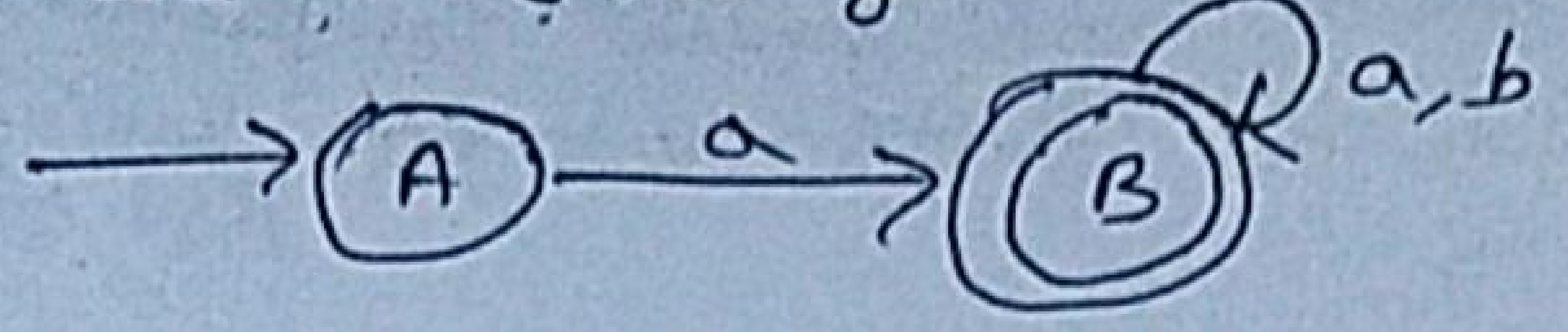
where L is head move left and R is head move right.

q_0 : Initial stage.

F : Set of final or halting states, accepting or rejecting states.

(2018) **Broad Questions**

3.) a) Convert the following NFA in its equivalent DFA.



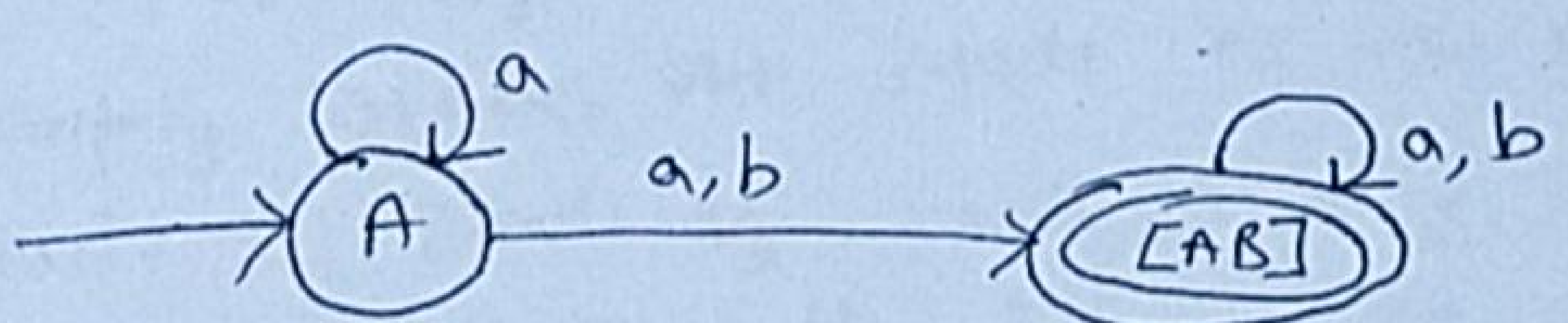
→ Transition table for the given NFA. —

States	a	b
A	B	ϕ
B	B	B

Transition table for its equivalent DFA —

States	a	b
A	A	[A B]
[AB]	[AB]	[AB]

Transition diagram for it's equivalent DFA —

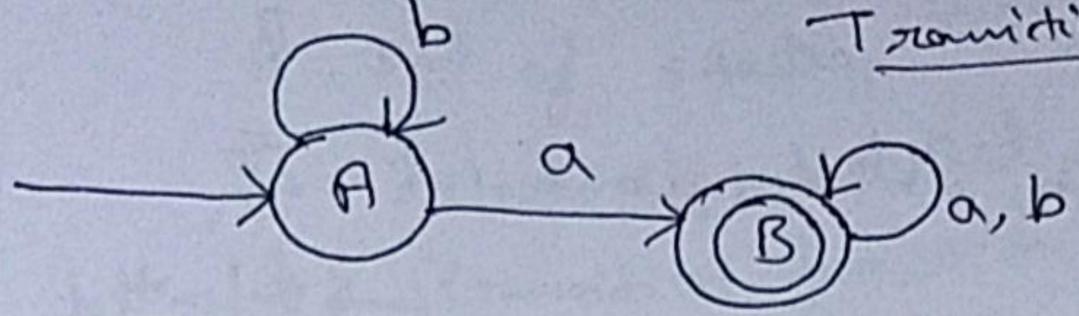


b) Construct a minimal DFA. which accepts set of all strings over {a, b}, where each string contains at least one 'a'.

→ $\Sigma = \{a, b\}$.

$L = \{a, ab, ba, aa, aba, abb. \ldots\}$ → Infinite set.

Transition Diagram.



Transition table:

States	a	b
A	B	A
B	B	B

$Q : \{A, B\}$

$\Sigma : \{a, b\}$

$\sigma(A, a) : B$, $\sigma(A, b) : A$, $\sigma(B, a) : B$, $\sigma(B, b) : B$.

$q_0 : A$

$F : \{B\}$.

2.) a) Give the formal definition of DFA.

→ Refer to the question 2(a) of 2017.

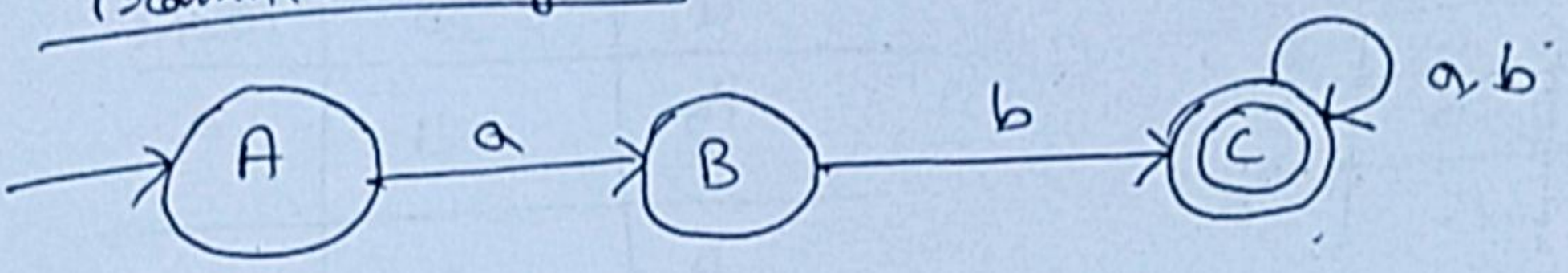
b) Construct and explain a NFA where it accepts all strings over {a,b}, where every string starts with 'ab'.

→ Σ = {a, b}

L = {ab, abba, abab, abbb, abaa, ...}.

→ Infinite set.

Transition diagram:



Transition table:

States	a	b
A	B	∅
B	∅	C
C	C	C

Q : {A, B, C}

Σ : {a, b}

δ(A,a) : B, δ(A,b) : ∅, δ(B,a) : ∅, δ(B,b) : C,

δ(C,a) : C, δ(C,b) : C

q_0 : {A}

F : {C}.

4.)

a) Give the formal definition of Moore machine.

→ It is a finite state machine where output depends on the current state only.

It is represented by 6-tuples $(Q, \Sigma, \delta, q_0, \Delta, \lambda)$.

Q: set of states.

Σ: Input alphabet set.

δ: Transition function which is represented by $Q \times \Sigma \rightarrow Q$.

q_0: Initial state.

Δ: Output alphabet set.

λ: Output function which is represented by $Q \rightarrow \Delta$.

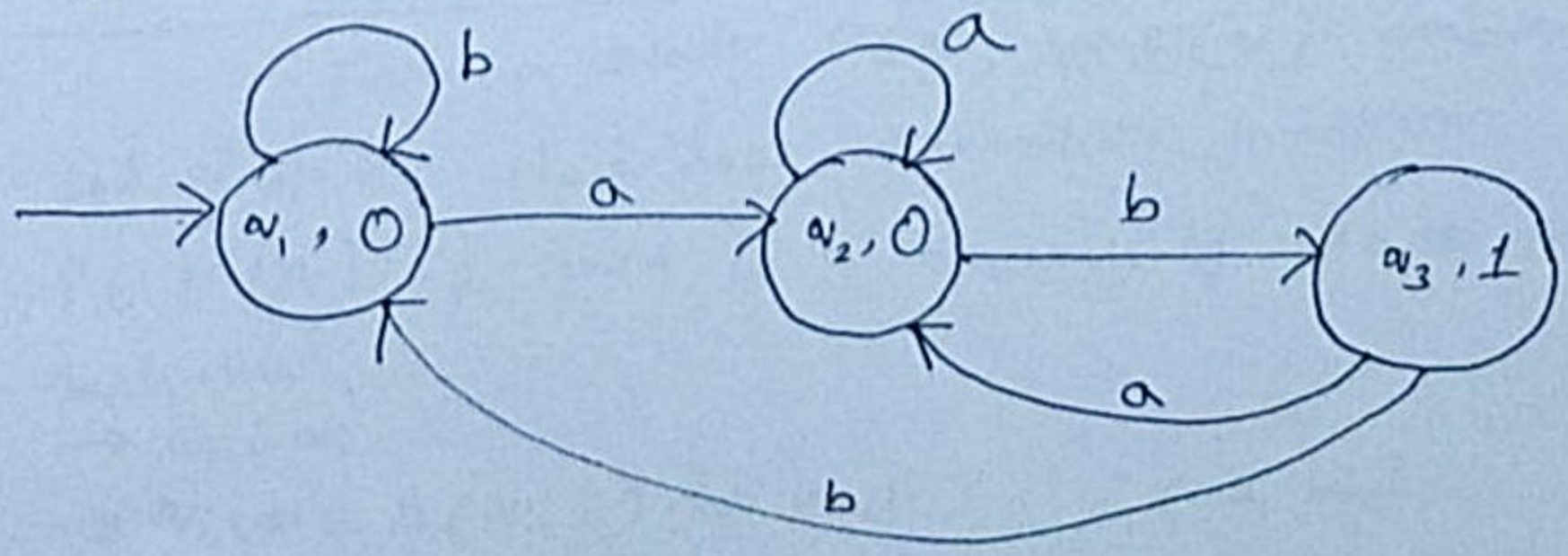
b) Construct a Moore machine that takes set of all strings over $\{a, b\}$ as inputs and prints '1' as an output for every occurrence of 'ab' as a substring.

→ Let, $E = \{a, b\}$.

and, $\Delta = \{0, 1\}$.

where E and Δ are the input and output alphabets respectively.

The required Moore Machine:



5.) a) Consider the following Mealy machine which has the input alphabet $\Sigma = \{a, b\}$ and the output alphabet $\Delta = \{0, 1\}$. Convert to its equivalent Moore machine.

State	a	b
q_0	$q_3, 0$	$q_1, 1$
q_1	$q_0, 1$	$q_3, 1$
q_2	$q_2, 1$	$q_2, 0$
q_3	$q_1, 0$	$q_0, 1$

→

b)

→

∴ Reg

6.) a) Co

→

He
∈
∈-c

set
PE
suc
→
→
→

b) Given, $\Sigma = \{a, b\}$. Find out the regular expression for the set of all strings whose length is even.

⟶ $\Sigma = \{a, b\}$

$L = \{aa, ab, ba, bb, aabb, abab, baba, bbaa. \ldots\}$

∴ Regular Expression $= (aa + ab + ba + bb)^*$

6.) a) Give the formal definition of ϵ - NFA.

⟶ An ϵ-NFA is represented by 5 tuples $(Q, \Sigma, \Delta, q_0, F)$ where,

Q : Finite set of states.

Σ : Finite set of input symbols / alphabets.

Δ : Transition function which is represented by —

$Q \times (\Sigma \cup \{\in\}) \rightarrow P(Q)$.

q_0 : Initial state.

F : Set of states distinguished as accepting states $F \subseteq Q$.

Here, $P(Q)$ denotes the power set of Q and $\in$ denotes the empty string.

ϵ-closure of a state :

For a state $P \in Q$, let $E(q)$ denotes the set of states that have transition functions Δ, i.e. $P \in E(q)$ if there is a sequence of states $q_1, \ldots q_k$. such that,

→ $q_1 = q_i$

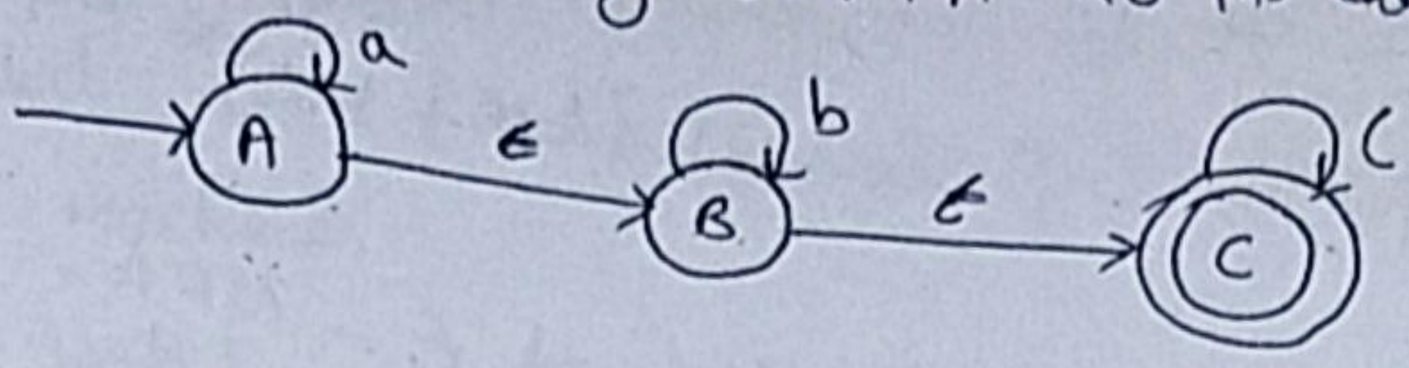
→ $q_{i+1} \in \Delta(q_i, \epsilon)$ for each $1 \leq i < k$ and.

→ $q_k = p$

$E(q)$ is known as the ϵ-closure of q. ϵ-closure is also defined for a set of states. The ϵ-closure of a set of states, p of an NFA is defined as the set of states reachable from any state in p ϵ-transition.

Formally, for $P \leq Q$, $E(P) = U_{q \in P} E(q)$

b) Convert the following ϵ-NFA to its equivalent NFA.



$\longrightarrow$

Let, New set of states be ——

Q' : {ϵ-closure (A), ϵ-closure (B), ϵ-closure (C)}

= {{AB}, {BC}, {C}}.

Q' : {q_a, q_b, q_c} → new States of NFA.

where,

q_a = {AB}, q_b = {BC}, q_c = {C}

Transition table for NFA ——

States	a	b	c
q_a	q_b	{ϕ}	{ϕ}
q_b	{ϕ}	q_b	{ϕ}
q_c	{ϕ}	{ϕ}	q_c
{ϕ}	{ϕ}	{ϕ}	{ϕ}

∴ Transition function ——

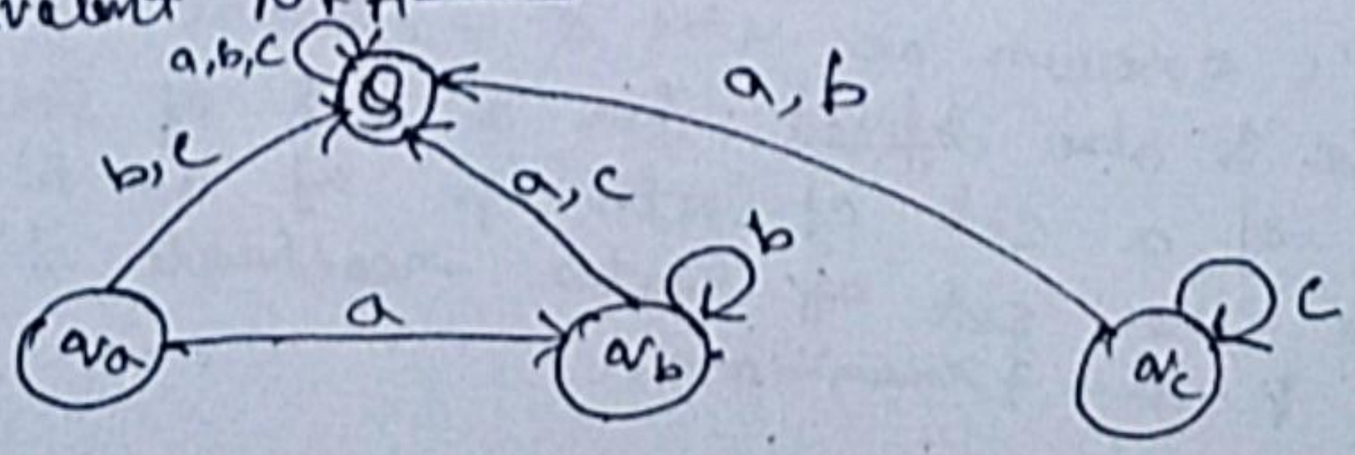
$\delta(q_a, a)$: q_b, $\delta(q_a, b)$: {ϕ}, $\delta(q_a, c)$: {ϕ}, $\delta(q_b, a)$: {ϕ},

$\delta(q_b, b)$: q_b, $\delta(q_b, c)$: {ϕ}, $\delta(q_c, a)$: {ϕ},

$\delta(q_c, b)$: {ϕ}, $\delta(q_c, c)$: q_c, $\delta(\{\phi\}, a)$: {ϕ},

$\delta(\{\phi\}, b)$: {ϕ}, $\delta(\{\phi\}, c)$: {ϕ}.

∴ Equivalent NFA ——



7) a) State and prove the Arden's theorem.

→ The Arden theorem states that:

$R = Q + \overline{R}P$.

Let, P and Q are two regular expressions and P does not contain ϵ, the epression $R = Q + \overline{R}P$; is represented by $R = QP^*$.

Proof:

$R = Q + \overline{R}P$.

$= Q + (Q + RP)P$

$= Q + QP + RP^2$

$= Q + QP + P^2(Q + RP)$

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

$= Q(\epsilon + P + P^2 + \ldots)$

$= QP^*$ (proved).

b) Prove that $(1 + 00^*1) + (1 + 00^*1)(0 + 10^*1)^*(0 + 10^*1)$ is equal to $0^*1(0 + 10^*1)^*$.

→ L.H.S. $= (1 + 00^*1) + (1 + 00^*1)(0 + 10^*1)^*(0 + 10^*1)$

$= (1 + 00^*1)(\wedge + (0 + 10^*1)^*(0 + 10^*1))$

$= (\wedge + 00^*)1(0 + 10^*1)^*$

$= 0^*1(0 + 10^*1)^* =$ R.H.S. [Proved].

∴ {07}

8.)
a) write a short note on simplification of CFG.

→ In a CFG, it may happen that all the productions rules and symbols are not needed for the derivation of strings. Besides, there may be some null productions and unit productions. Elimination of these productions and Symbols is called simplification of CFGs.
Simplification essentially comprises of the following steps—

- Reduction of CFG
- Removal of Unit Productions
- Removal of Null Productions.

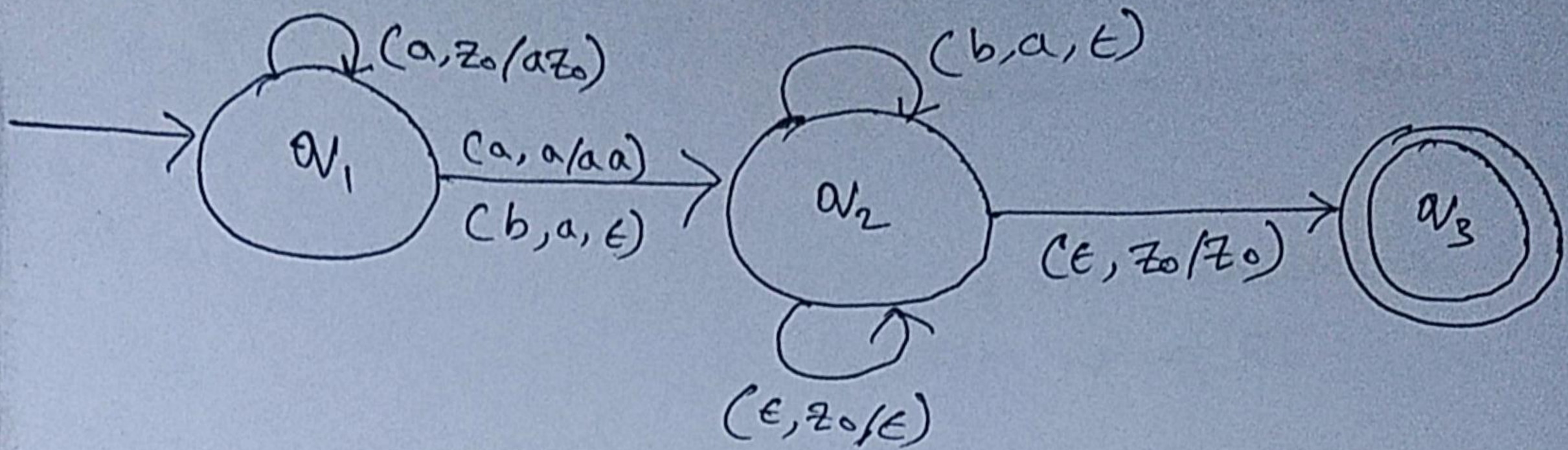
b) $G = \{(S, A, B), (a), P: S \rightarrow AB \mid a, A \rightarrow 4a\}$.
Simplify the grammar.

→
$\Rightarrow S \rightarrow AB \mid a$
$A \rightarrow 4a$.
we cannot obtain a set of Context Free Langugage from the given gramas as if we use the following dependencies, the corresponding limitations would be encountered.
$\varphi \rightarrow AB \mid a$.
$S \rightarrow AB \mid a$
$A \rightarrow 4a$
So, we don't have any terminals for B.
Hence, to get a set for Context language for the given grammar, we need terminal for B.

9.)
a)
→
b) Des
lan
→
1→1
↳→L

9.
a) Construct a PDA that accepts a language,
$L = \{a^n b^n \mid n \geq 1\}$.

→



b) Design a turing machine which recognizes the language $L = 01^*0$.

b = ␣

→ $\Sigma = \{0, 1\}$

$L = 01^*0$.