Assignment-1

Q1) a) What are the key differences between complexity theory, computability theory, and automata theory, and automata theory and how do their concepts interrelate in defining the limits of computation?

Ans → i) Complexity Theory →

a) Studies resource requirements (time, space) for problem-solving.

b) Focuses on classifying problems by how efficiently they can be solved.

c) Example → Deciding if a problem can be solved in polynomial time or requires exponential time.

ii) Computability Theory →

a) Examines the limits of what is computable at all, ignoring resource constraints.

b) Differentiates decidable problem from undecidable ones.

c) Example → The halting problem shows there's no universal algorithm to decide if any given program will halt or run forever.

iii) Automata Theory →

a) Analyzes abstract computational models (automata) and the types of languages they can recognize.

b) Classifies languages by computational ~~models~~ power needed, ~~i~~ e.g. finite automata for regular languages.

c) Example → Finite automata can recognize patterns in regular languages, but can't handle more complex structures.

b) What are the differences between an alphabet, a string, and a language in automata theory? Include examples to illustrate these concepts.

Ans → i) Alphabet (Σ) → A set of symbols. Eg → For binary, $\Sigma = \{0,1\}$

ii) String → A finite sequence of symbols from an alphabet. Eg → With $\Sigma = \{a,b\}$, "aab" is a string over Σ.

iii) Language → A set of strings over an alphabet, usually with certain rules. Eg → For

$\Sigma = \{0,1\}$, a language could be "all binary strings with an even number of 0's".

c) Differentiate between the Kleene star closure (Σ^*) and Positive closure (Σ^+) with suitable examples.

Ans → i) Kleene Star (Σ)*: Set of all possible strings over an alphabet Σ. Eg → With $\Sigma = \{a,b\}$, Σ^* includes "", "a", "b", "aa", etc.

ii) Positive Closure (Σ^+): Set of all non-empty strings over Σ (excludes the empty string).
Eg → With $\Sigma = \{a,b\}$, Σ^+ includes "a", "b", "aa", but not "".

2) a) Let $S(n) = 1+2+3+\cdots+n$ be the sum of the first n natural numbers and let $C(n) = 1^3+2^3+3^3+\cdots+n^3$ be the sum of the first n cubes. Prove the following equalities by induction on n, to arrive at the curious conclusion that $C(n) = S^2(n)$ for every n.

i) $S(n) = \frac{1}{2}n(n+1)$

ii) $C(n) = \frac{1}{4}(n^4+2n^3+n^2) = \frac{1}{4}n^2(n+1)^2$

Ans → i) $S(n) = \frac{1}{2}n(n+1)$ by induction

a) Base case → For $n=1$,

$$S(1) = 1 = \tfrac{1}{2}\cdot 1\cdot(1+1) = 1$$

b) Induction Hypothesis → Assume the formula holds for k, i.e.,

$$S(k) = \tfrac{1}{2}k(k+1)$$

c) Induction Step → Prove it for $k+1$,

$$S(k+1) = S(k) + (k+1)$$

$$\therefore S(k+1) = \tfrac{1}{2}k(k+1) + (k+1)$$

$$= (k+1)(\tfrac{1}{2}k+1) = \tfrac{1}{2}(k+1)(k+2)$$

$\therefore S(n) = \frac{1}{2}n(n+1)$ is proven.

ii) $C(n) = \frac{1}{4}(n^4+2n^3+n^2) = \frac{1}{4}n^2(n+1)^2$ by induction

a) Base Case: For $n=1$,

$$C(1) = 1^3 = 1 = \tfrac{1}{4}(1^2)(1+1)^2 = 1$$

b) Induction Hypothesis → $C(k) = \frac{1}{4} k^2 (k+1)^2$

c) Induction Step → $C(k+1) = C(k) + (k+1)^3$

$$= \frac{1}{4} k^2 (k+1)^2 + (k+1)^3$$

$$= (k+1)^2 \left(\frac{1}{4} k^2 + k + 1\right) = \frac{1}{4} (k+1)^2 (k^2 + 4k + 4)$$

$$= \frac{1}{4} (k+1)^2 (k+2)^2$$

∴ By mathematical induction, $C(n) = \frac{1}{4} n^2 (n+1)^2$ is proven.

b) Proof by contradiction that $\sqrt{5}$ is irrational.

Ans → Assume $\sqrt{5}$ is a rational. Then there exists integers a and b (with $b \neq 0$) such that $\sqrt{5} = a/b$ and a/b is in simplest form.

Then, $\sqrt{5} = a/b$ implies $5 = a^2/b^2$, so $a^2 = 5b^2$.

This, means a^2 is divisible by 5. Therefore, a must be divisible by 5.

Let $a = 5k$, for some integer k. Then $(5k)^2 = 5b^2$, so $25k^2 = 5b^2$, and thus $5k^2 = b^2$

Hence, b^2 is also divisible by 5, implying b must be divisible by 5.

But if both a and b are divisible by 5, a/b is not in simplest form, which contradicts our original assumption.

Thus, $\sqrt{5}$ ~~cann~~ is irrational.

c) Show that every graph with ≥ 2 nodes contains two nodes that have equal degrees.

Ans → Let G be a graph with n nodes, where $n \geq 2$.

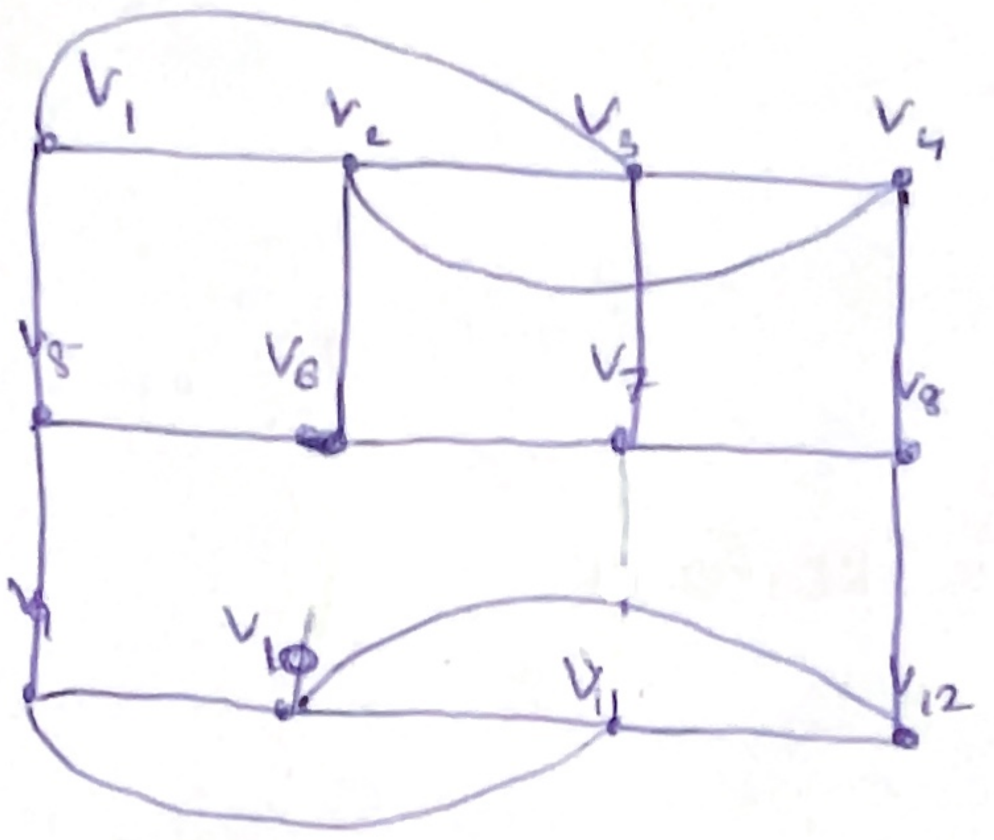
The maximum possible degree of any node is $n-1$ (if it is connected to every other nodes).

The degrees of nodes range from 0 to $n-1$. However, if one node has degree 0, another node cannot have degree $n-1$ ~~possible~~ because the isolated node would not connect to it. Therefore, the degrees of nodes range from 0 to $n-2$, providing n possible degrees. By the pigeonhole principle, in a set of n nodes and $n-1$ possible degrees, at least two nodes must share the same degree.

Name: ______________ Regd. Number: ______________

3) a) A graph G is said to be k-regular if every node in the graph has degree k. Construct a 3-regular graph $G = (V, E)$ with 12 nodes. Display the vertex set V and edge set E of the graph G.

Ans→ G is k-regular, if every node in the graph has degree k.



Degree of all vertices is 3, so graph G is 3-regular with 12 nodes.

Vertex Set → $V = \{V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9, V_{10}, V_{11}, V_{12}\}$

Edge Set → $\{(V_1, V_2), (V_1, V_5), (V_1, V_3), (V_2, V_3), (V_2, V_4), (V_2, V_6), (V_3, V_4), (V_3, V_7), (V_4, V_8), (V_5, V_6), (V_5, V_9), (V_6, V_7), (V_7, V_8), (V_8, V_{12}), (V_9, V_{10}), (V_9, V_{11}), (V_{10}, V_{12}), (V_{10}, V_{11}), (V_{11}, V_{12})\}$

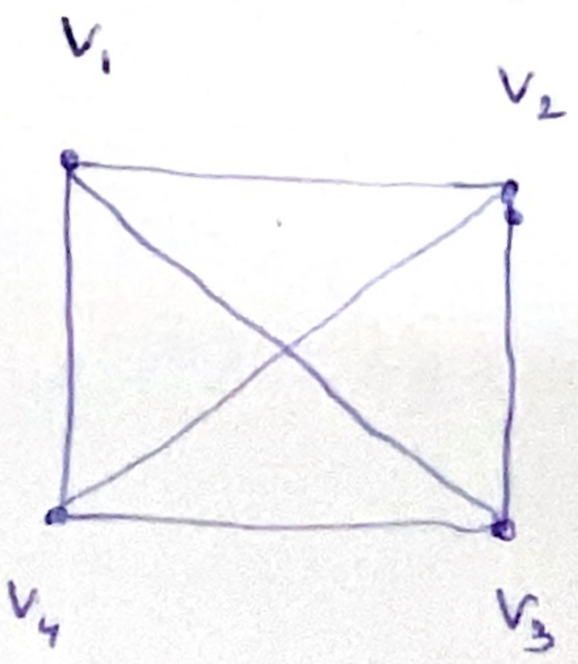
b) For a k-regular graph, if k is odd, then the number of vertices of the graph must be even. True/False? Justify your answer by constructing a suitable graph.

Ans → For a k-regular graph, every vertex have k degree.

So, $\sum_{v \in V} \deg(v) = 2|E| \Rightarrow nk = 2|E|$

As the no. of edge is multiple of 2.

Let's construct a simple 3-regular graph with n is even. $V = \{V_1, V_2, V_3, V_4\}$



6 no. of edges.

If n is odd then $|E|$ is multiple of 2 which ~~construct~~ contradicts. So for the graph to be k-regular where k is odd, then the number of vertices n must be indeed be even.

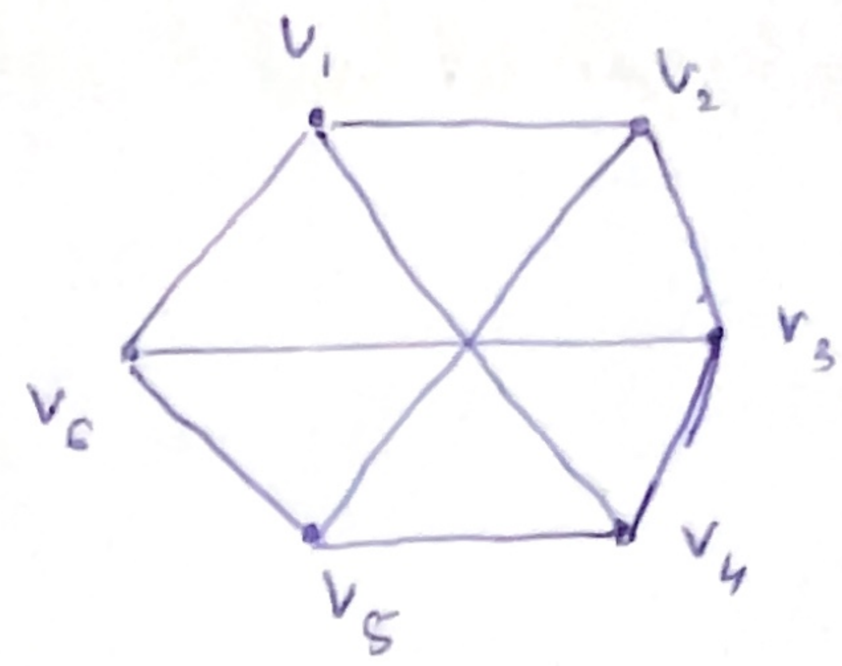
c) Using proof by construction, show that "For each even number $N >= 2$, there exist a 3-regular graph with N vertices".

Ans→ Constructing a 3-Regular graph for $N = 6$

Vertices, $V = \{V_1, V_2, V_3, V_4, V_5, V_6\}$

Cycle edges : $(V_1, V_2), (V_2, V_3), (V_3, V_4), (V_4, V_5), (V_5, V_6), (V_6, V_1)$

Chord Edges → (V_1, V_4), (V_2, V_5), (V_3, V_6)



This gives us a 3-regular graph where each vertex is connected to exactly 3 other vertices, as required.

d) Number of edges of a K-regular graph with N vertices is $|E| = (N*K)/2$. Write down the formula to construct the edges of a 3-regular graph.

Ans → Given,

$$|E| = \frac{N \times K}{2}$$

For a 3-regular graph → $|E| = \frac{N \times 3}{2} = 3N/2$. Since N must be even for a 3-regular graph, this result will always yield an integer value for $|E|$.

Edge Construction Formula a 3-Regular Graph →

i) Vertices → Label the vertices as $V = \{v_1, v_2, \dots, v_n\}$

ii) Cycle Edges → Connect each vertex v_i to its next neighbor in a cycle: (v_i, v_{i+1}) for $i = 1, 2, \dots, N-1$, and (v_N, v_1)

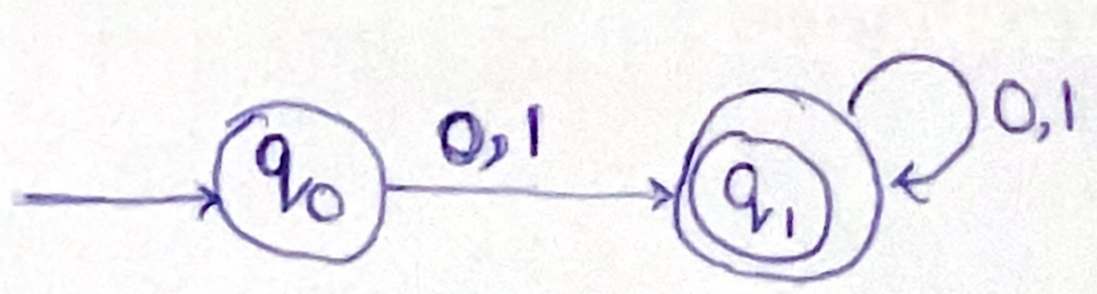
iii) Chord edges → Connect each vertex v_i to the vertex $v_{i+N/2}$ (mod N) which is the vertex halfway around the cycle:

$$(v_i, v_{(i+N/2) \bmod N}) \quad \text{for } i = 1, 2, \dots, N$$

4) Design the DFA's recognizing the following languages over input alphabets $\Sigma = \{0,1\}$

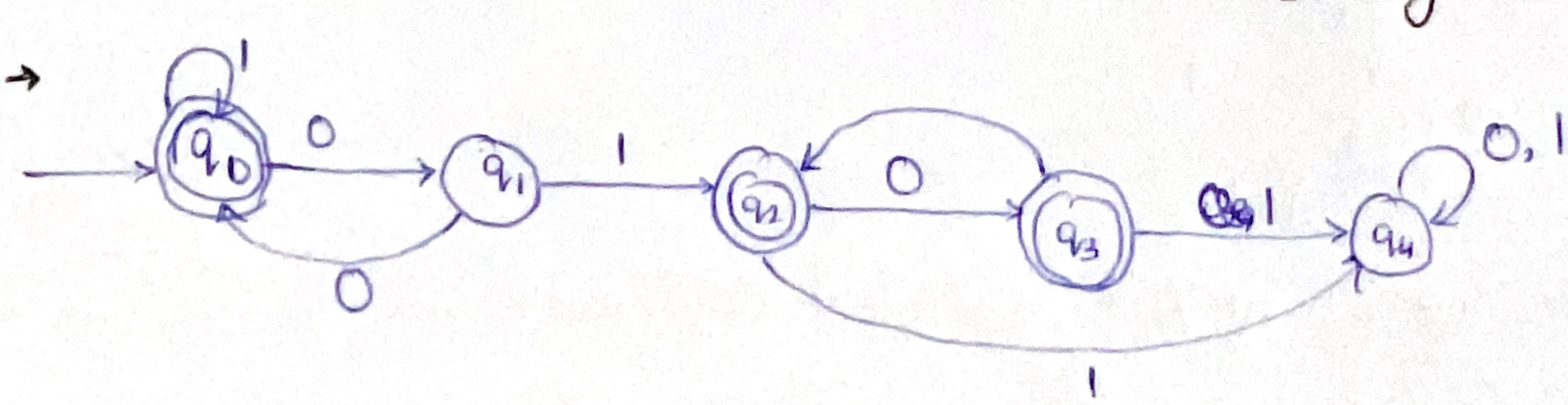
a) {All strings except the empty string}

Ans →



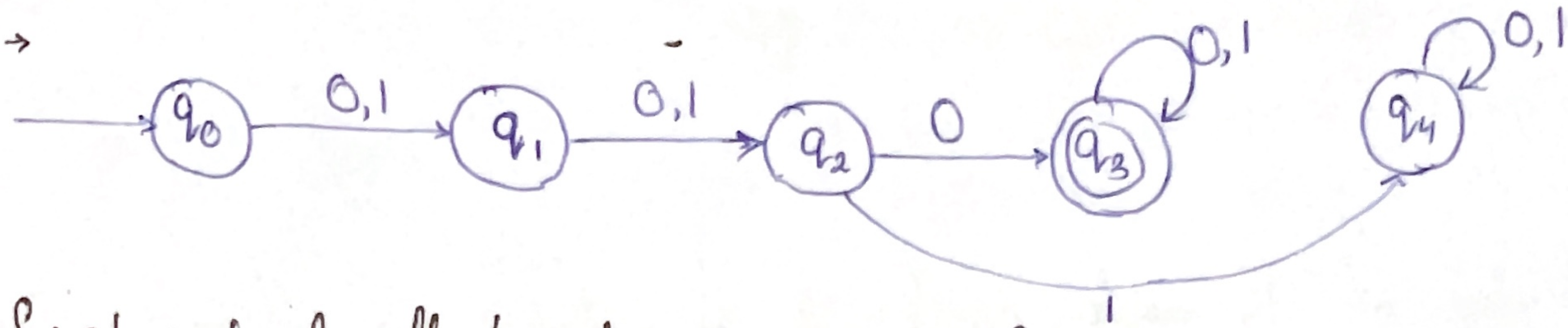
b) {w | w contains an even number of 0s or contains exactly two 1s}

Ans →



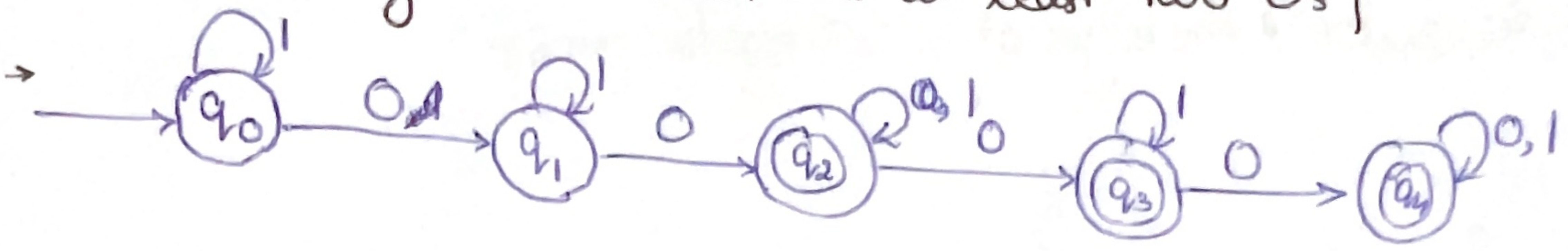
c) $\{w \mid w$ has length at least 3 and its third symbol is a $0\}$

Ans →



d) $\{w \mid w$ has length 4 and contains at least two 0s$\}$

Ans →

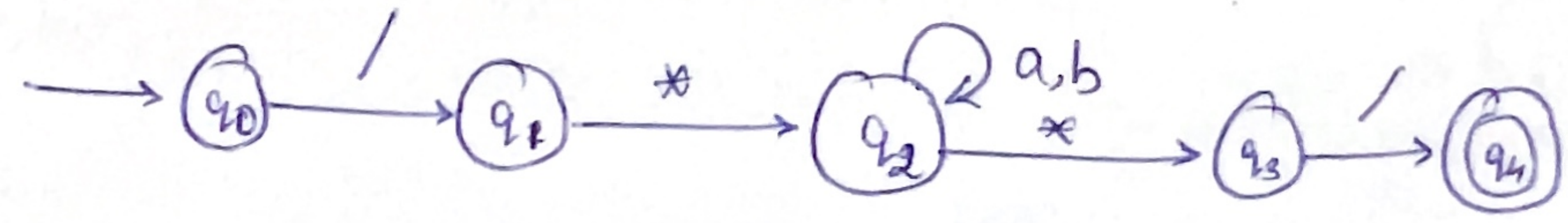


5) In certain programming languages, comments appear between delimiters such as /* and */. Let C be the language of all valid delimitted comment strings. A member of C must begin with /* and end with */ but have no intervening */. For simplicity, assume that the alphabet for C is $\Sigma = \{a, b, /, *\}$

a) Construct a finite automata that recognizes C.

b) Define a regular expression that generates C.

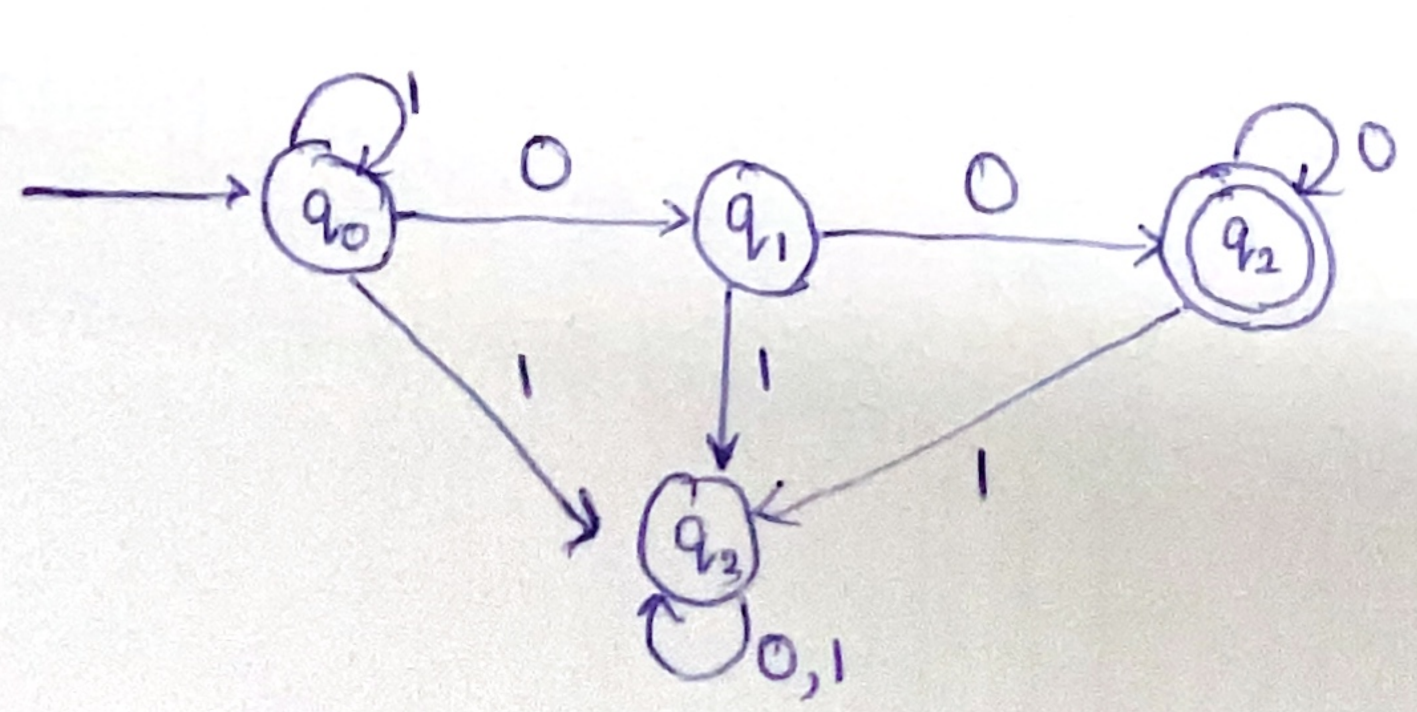
Ans → a)



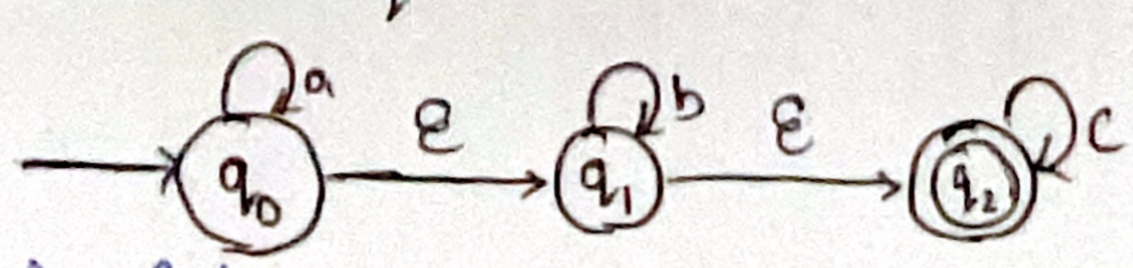
b) Regular Expression → $/*\,(a+b)^*\,*/$

6) a) Design a NFA that accepts the language $L = \{w \mid w$ contains at least two 0's and at most one 1 over input alphabets $\Sigma = \{0, 1\}$.

Ans →



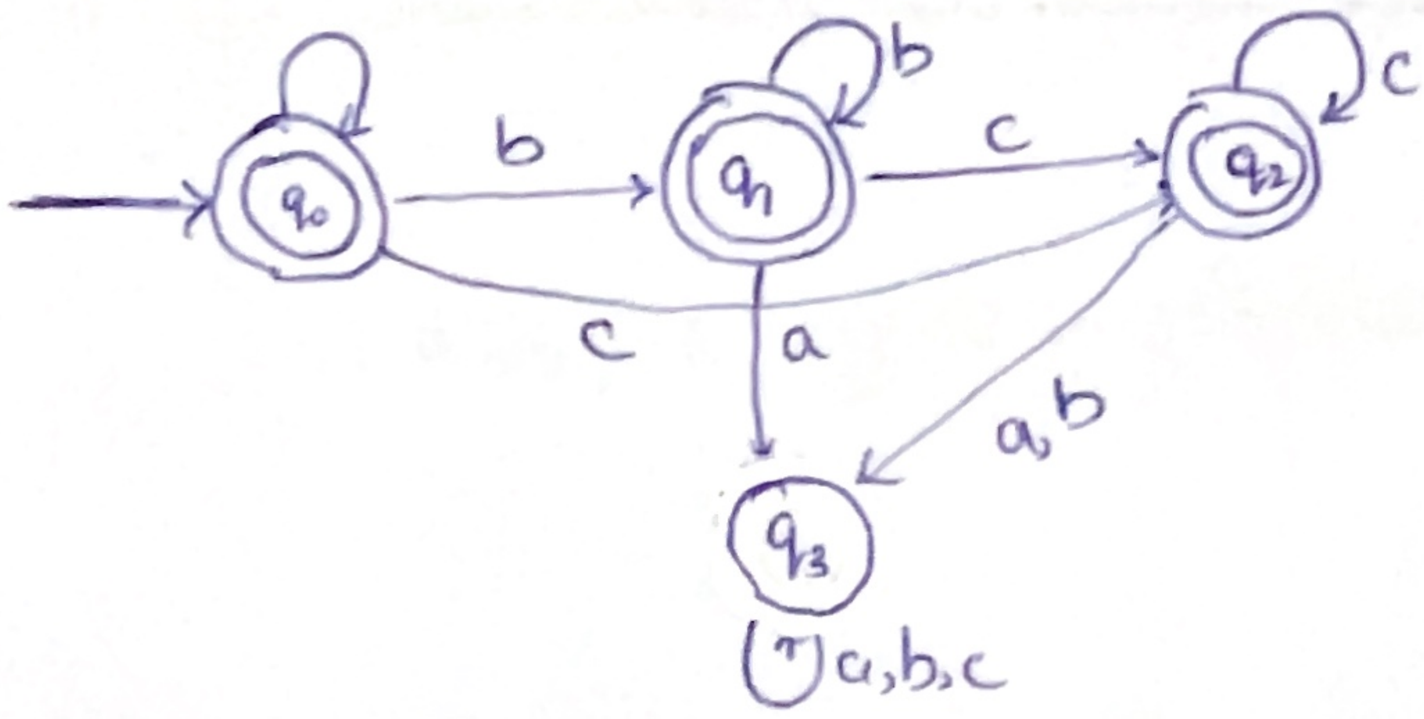
b) Convert the following ϵ-NFA to equivalent DFA and identify the language that is recognized by the DFA.



Ans → $\delta(q_0, a) = \{q_0\}$ $\quad \delta(q_1, a) = \{\phi\}$ $\quad \delta(q_2, a) = \{\phi\}$

$\delta(q_0, b) = \{\phi\}$ $\quad \delta(q_1, b) = \{q_1\}$ $\quad \delta(q_2, b) = \{\phi\}$

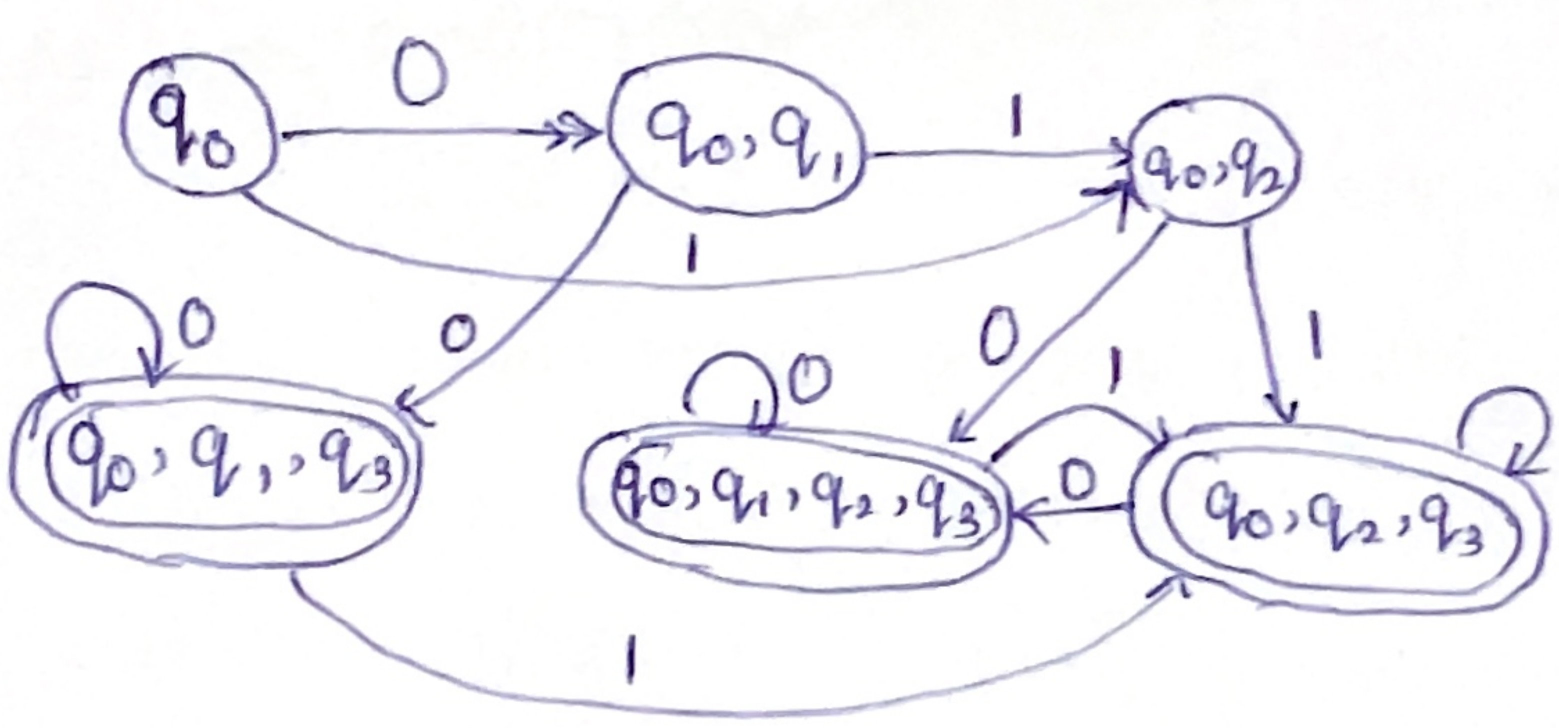
$\delta(q_0, c) = \{\phi\}$ $\quad \delta(q_1, c) = \{\phi\}$ $\quad \delta(q_2, c) = \{q_2\}$



c) Design a DFA for the transition table as given below and determine the language that is recognized by the DFA.

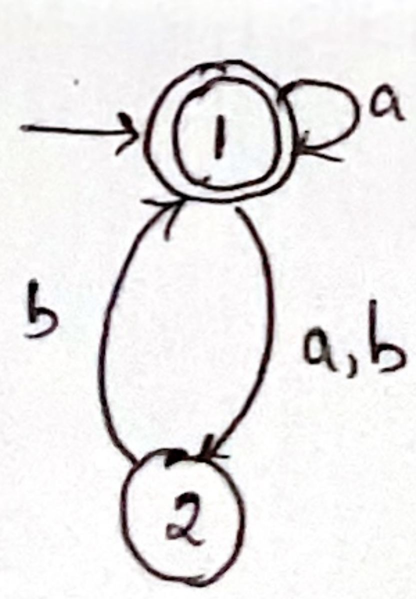
Present State	0	1
$\rightarrow q_0$ (initial)	$\{q_0, q_1\}$	$\{q_0, q_2\}$
q_1	$\{q_3\}$	ϕ
q_2	$\{q_2, q_3\}$	$\{q_3\}$
q_3 (final)	$\{q_3\}$	$\{q_3\}$

Ans→

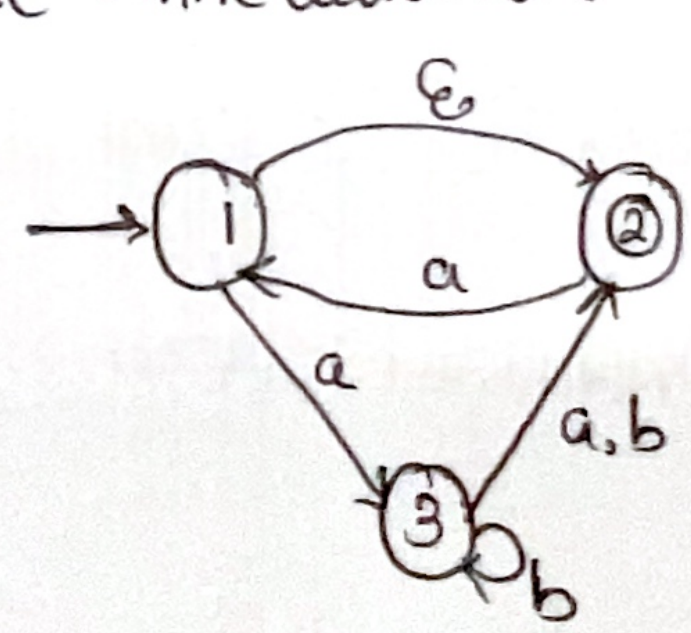


7) Use ~~that~~ state construction method to convert the following two non-deterministic finite automata to equivalent deterministic finite automata.

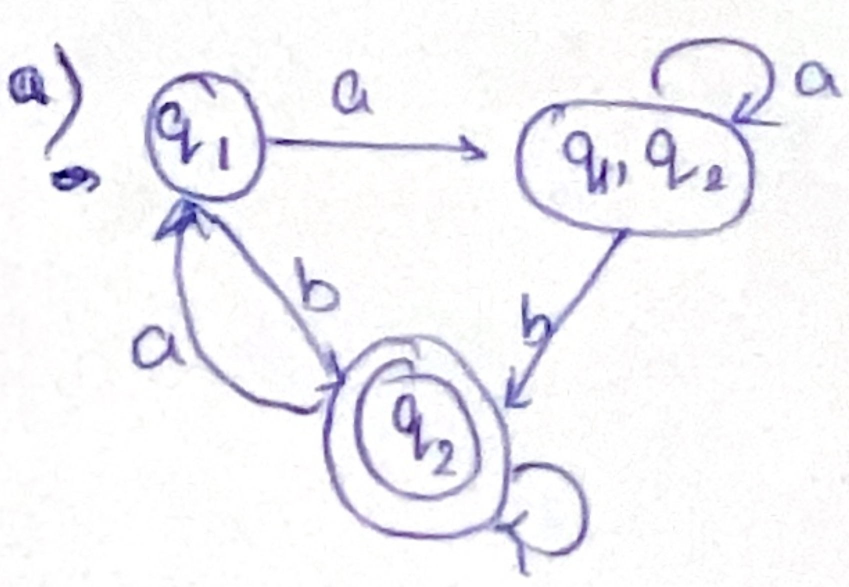
a)



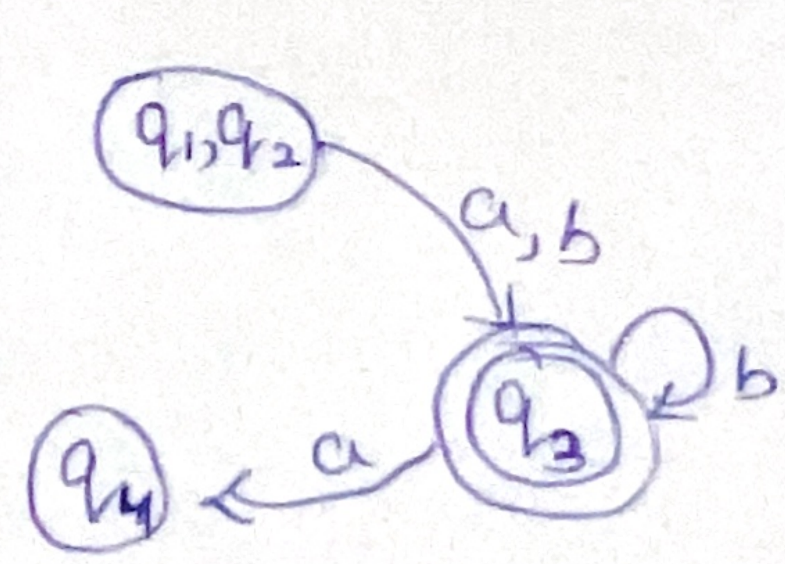
b)



Ans→ a)



b)



Q8) Let $\Sigma = \{a, b\}$

i) Write regular expression to define language consisting of strings 'w' such that, 'w' contains only a's or only b's of length zero or more.

Ans → RE → $(a^* + b^*)$

ii) Write regular expression to define language consisting of strings 'w' such that, 'w' is of length one or more and contains only a's or only b's.

Ans → RE → $(a^+ + b^+)$ Here, a^+ & b^+ respresents one or more a's and b's respectively.

iii) Write regular expression to define language consisting of strings 'w' such that, 'w' of length odd containing only b's.

Ans → RE → $b(bb)^*$

iv) ...., 'w' ~~of length~~ contains zero or more a's followed by zero or more b's.

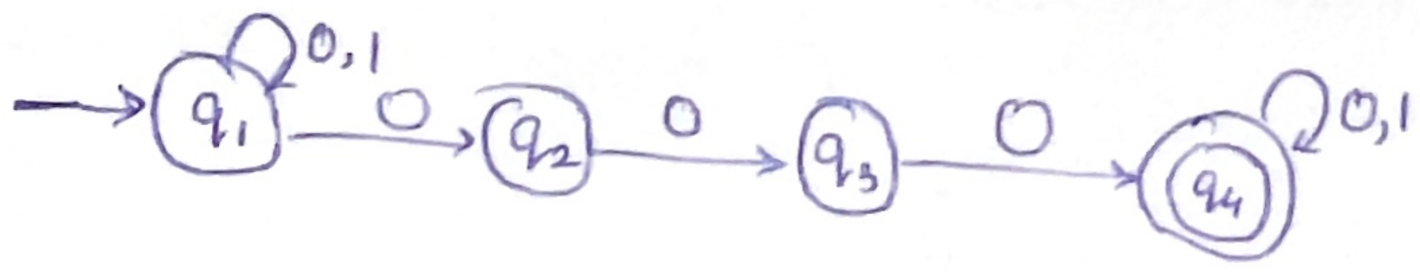
Ans → RE → $a^* b^*$

v) ......, 'w' always starting with a.

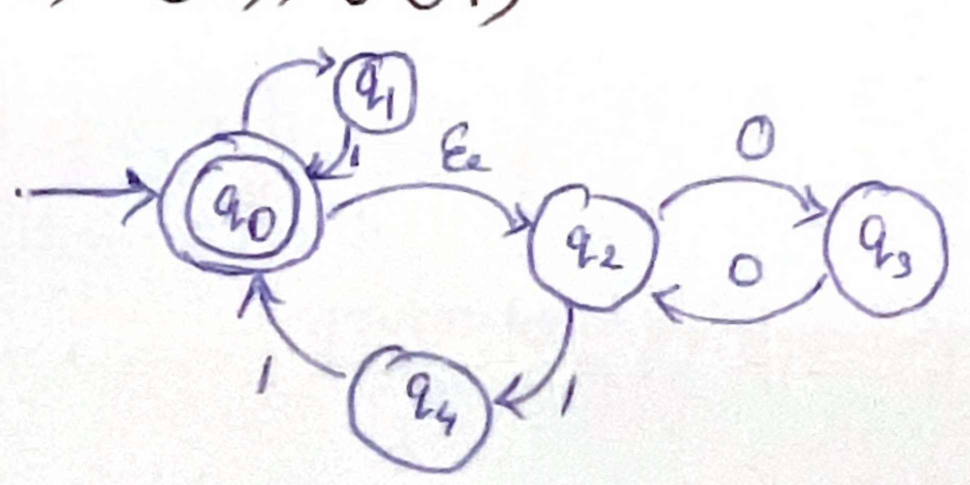
Ans → RE → $a(a+b)^*$

9) a) Convert the following regular expressions to non-deterministic finite automata.

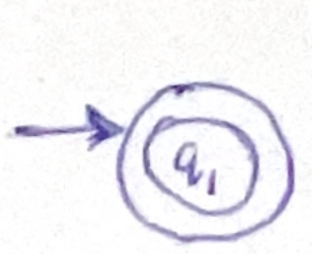
i) $(0 \cup 1)^* \, 000 \, (0 \cup 1)^*$

Ans →


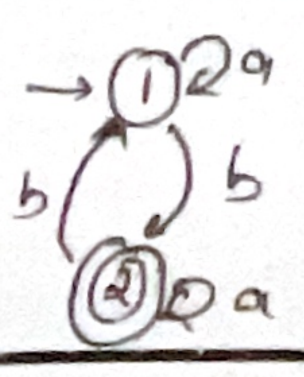
ii) $(((00)^* (11)) \cup 01)^*$

Ans →


iii) ϕ^*

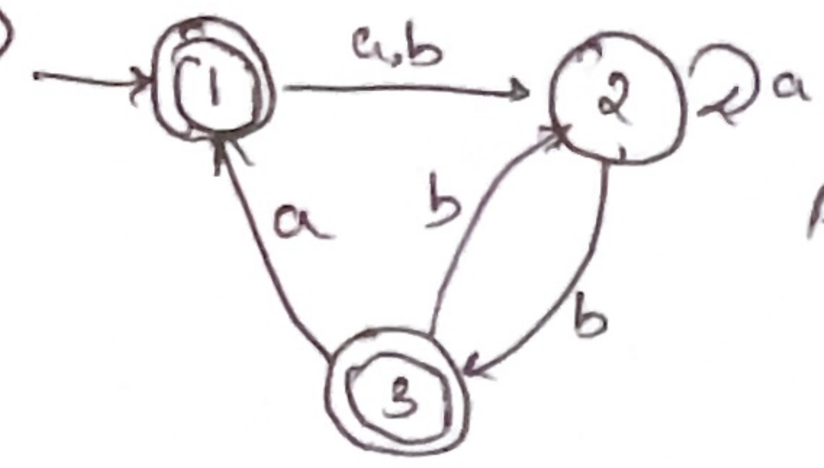
Ans →


b) Convert the following finite automatons to regular expressions.

i)


Ans → RE → ~~$a^* ba^* (bb)^*$~~ $(a^* ba^* b)^* a^*$

Name: ________________ Regd. Number: ________________

b) 

Ans → RE → $((a+b)a^*b(bb)^*)^*$

10) Using pumping lemma to show that the following languages are not regular.

a) $L1 = \{0^n 1^n 2^n \mid n \geq 0\}$

Ans → i) Assume L_1 is regular. Then by the Pumping Lemma, there exists a pumping length p such that any string s in L_1 with $|s| \geq p$ can be divided into three parts $s = xyz$ and pumped as described above.

ii) Choose a string s in L_1: Let $s = 0^p 1^p 2^p$, where $n = p$. Clearly, $|s| = 3p$, which satisfies $|s| \geq p$.

iii) Divide $s = xyz$ according to the Pumping Lemma: According to the Pumping Lemma, x and y must be within the first p symbols of s, because $|xy| \leq p$. This means that x and y consist only of 0s. Therefore, y contains only 0s and $|y| > 0$.

iv) Pumping y: Now, consider pumping y with $i = 2$. This would result in a new string & $s' = xy^2z$, which would have more than p 0s followed by p 1s and p 2s. Specially, s' would have than p 0s, exactly p 1's and exactly p 2s.

v) Check if $s' \in L_1$: The new string s' is not of the form $0^n 1^n 2^n$ because it has more 0s than 1s and 2s. Therefore, $s' \in L_1$, which contradicts the Pumping Lemma.

vi) Conclusion → Since pumpings leads to a string not in L_1, the language $L_1 = \{0^n 1^n 2^n \mid n \geq 0\}$ is not regular.

b) $L2 = \{www \mid w \in \{a,b\}^*\}$

Ans → i) Assume L_2 is regular. Then, by the Pumping Lemma, there exists a pumping length p such that any string s in L_2 with $|s| \geq p$ can be divided into three parts $s = xyz$ & pumped as described above.

ii) Choose a string s in L_2: Let $s = a^p b^p a^p b^p a^p b^p$, where $w = a^p b^p$. This string s is of the form www and has length $|s| = 6p$, which satisfies $|s| > p$.

iii) Divide $s = xyz$ according to the pumping Lemma; According to the Pumping Lemma, x and y must be within the first p symbols of s, because $|xy| \leq p$.

This means that x and y consist of a's, since the first p symbols of s are all a's. Therefore, y contains only a's and $|y|>0$.

iv) Pumping y: Now, consider pumping y with $i=2$. This would result in a new string $s' = xy^2z$, which would have more than p a's in the first segment, followed by p b's, p a's, pb's, pa's and p b's.

v) Check if $s' \in L_2$: The new string s' does not match the pattern wwww because the first segment has more than pa's, so the second and third segments do not match the first one. Therefore $s' \in L_2$, which contradicts the Pumping Lemma.

vi) Conclusion → Since pumping s leads to a string not in L_2, the language $L_2 = \{wwww \mid w \in \{a,b\}^*\}$ is not regular.