Module - 2

More on Regular Languages

Regular Expressions

→ Algebraic Representation of Regular Language

Operators of Regular Expression

1) Union
2) Concatenation
3) Kleene closure
4) Positive closure.

* $L = \{\varepsilon\}$ R.E corresponding to L is ε
* $L = \{a\}$ $R = a$ R → regular expression

$L = \{a, b\}$ $R = a \cup b = a + b$

$L = \{a^0, a^1, a^2, \dots\}$, $R = a^*$

$L_1 = \{a\}$ $R_1 = a$ $L_2 = \{b\}$ $R_2 = b$

→ ~~$R = R_1 + R_2$~~ Union -: ∪ or '+'

$L_1 \cup L_2 = \{a\} \cup \{b\} = \{a, b\}$

$L_1 = \{a\}$ · $R_1 = a$

$L_2 = \{b\}$ $R_2 = b$

$R = R_1 \cup R_2$
$= a \cup b$
$= a + b$ or a / b

→ Concatenation -:

$L_1 = \{a\}$ $R_1 = a$

$L_2 = \{b\}$ $R_2 = b$

$L_1 \cdot L_2 = \{ab\}$ $R = ab$

→ Kleene Closure -:

$L^* = \{a\}$ $\quad R = a$

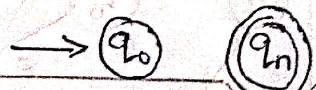
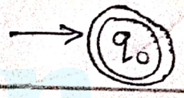
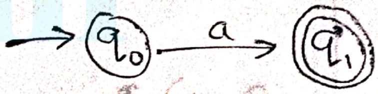
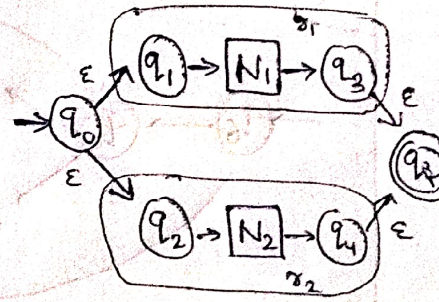
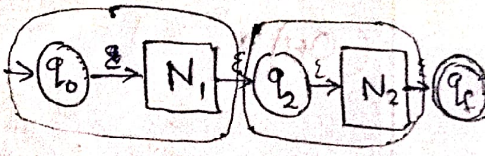
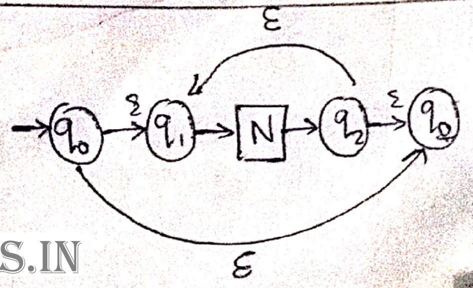
$L^* = \{a\}^*$ $\quad R^* = a^* = \{\varepsilon, a, aa, aaa, \ldots\}$

→ Positive Closure -:

$L = \{a\}$ $\quad R = a$

$L^+ = \{a\}^+$ $\quad R^+ = \{a, aa, \ldots\}$

$L^* - \varepsilon$

Regular Expressions	Regular set	Finite Automata
ϕ	{ }	
ε	$\{\varepsilon\}$	
Every a in Σ; a	{a}	
Let r_1 and r_2 be R.E then $(r_1 + r_2)$ or (r_1 / r_2)	$\{r_1 \cup r_2\}$	
$r_1 \cdot r_2$	R_1R_2 where R_1 and R_2 are regular set corresponding to r_1 and r_2	
r^*	R^* (where R is the regular set corresponding to r)	

Conversion of RE to FSA (Thompson Construction) :-

Q. Construct an ENFA for the regular expression

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

Ans $r = (0+1)^* 1 (0+1)$

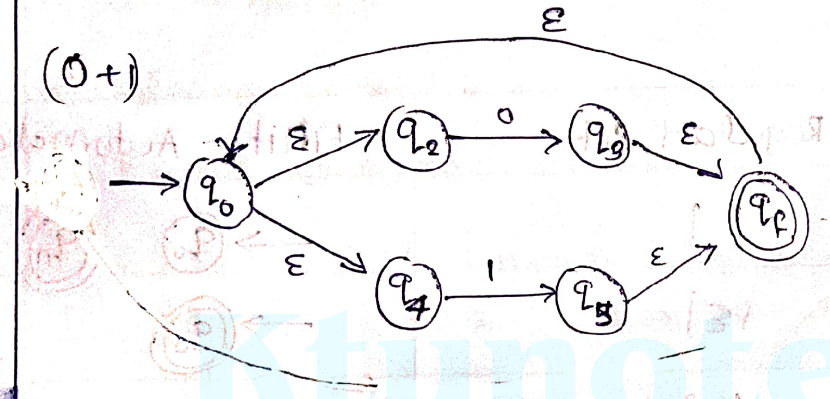
$r = r_1 \cdot r_2 \cdot r_3$

$r_1 = (0+1)^*$

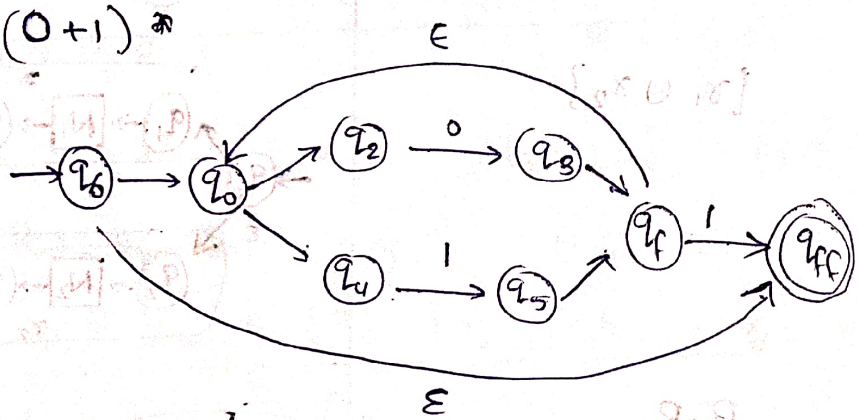
$r_2 = 1$

$r_3 (0+1)$

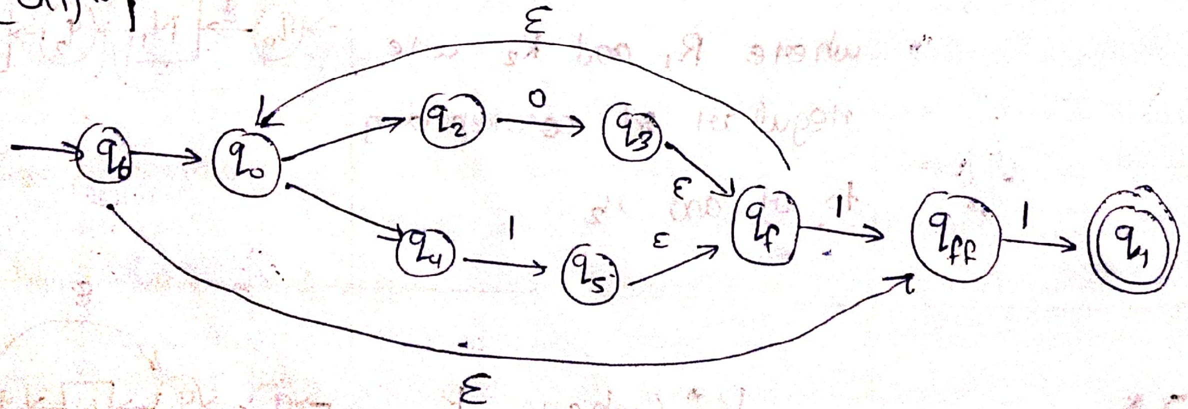
$(0+1)$



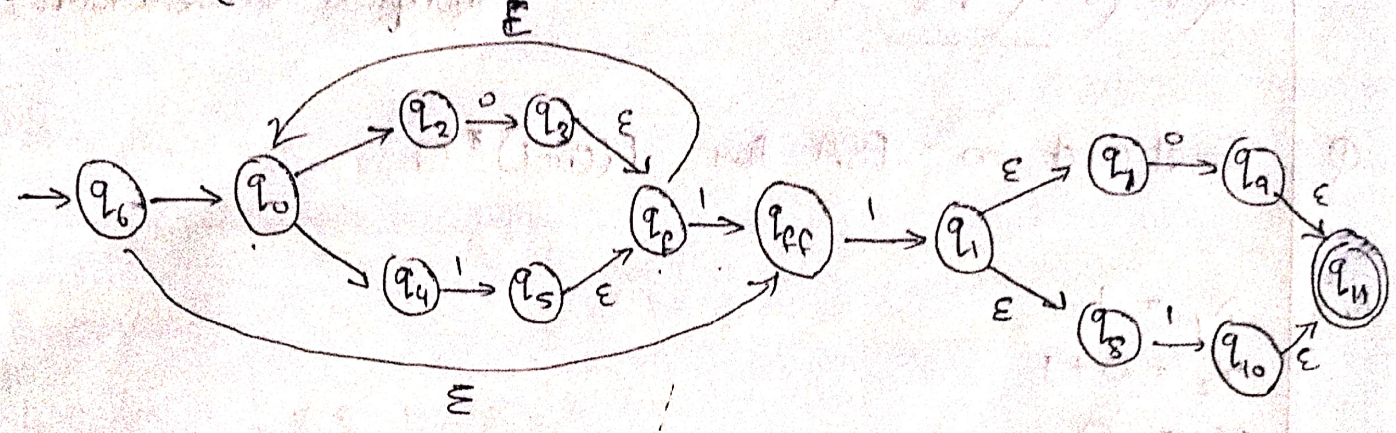
$(0+1)^*$



$(0+1)^* 1$

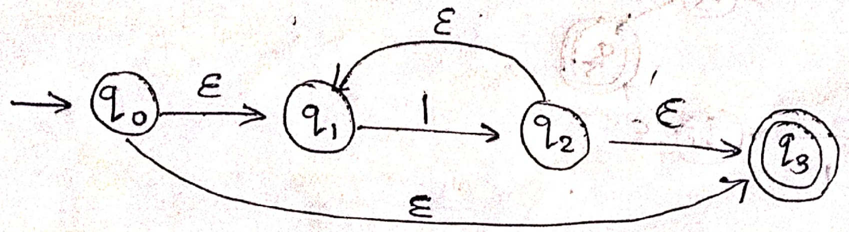


$(0+1)^*.1(0+1)$



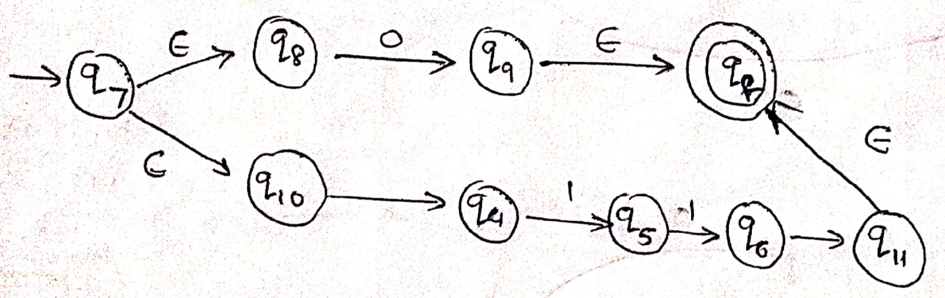
Q $1^*(0+11)1$

1^*

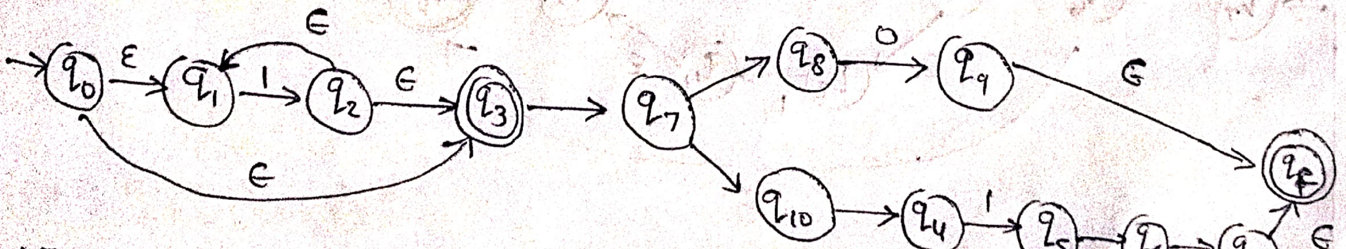


1.1

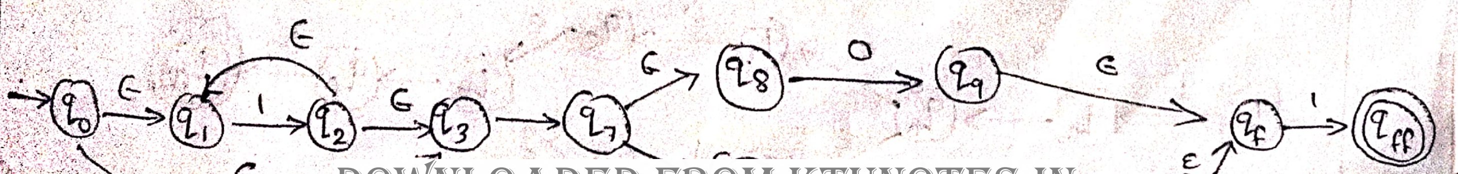
$(0+11)$



$1^*.(0+11)$



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



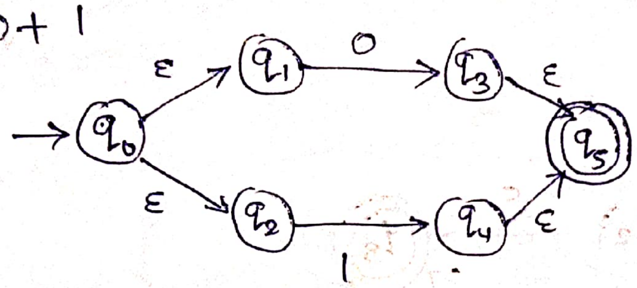
Q Construct an FSA for $1(0+1)^*0$

Ans $r_1 = 1$

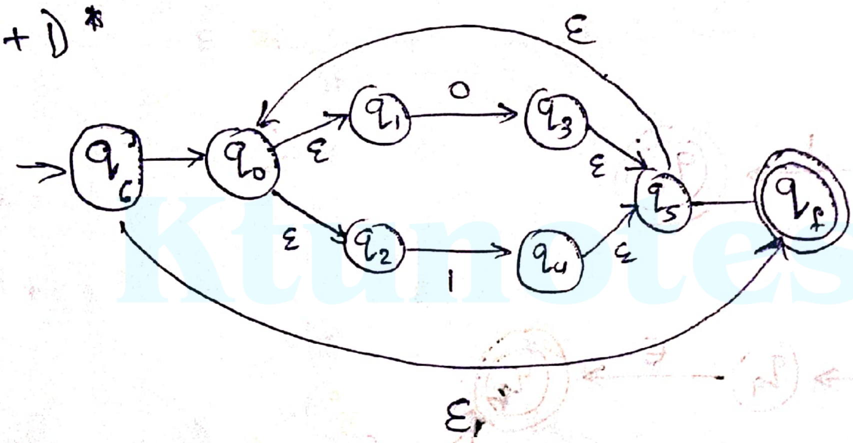
$r_2 = 0+1$

$r_3 = 0$

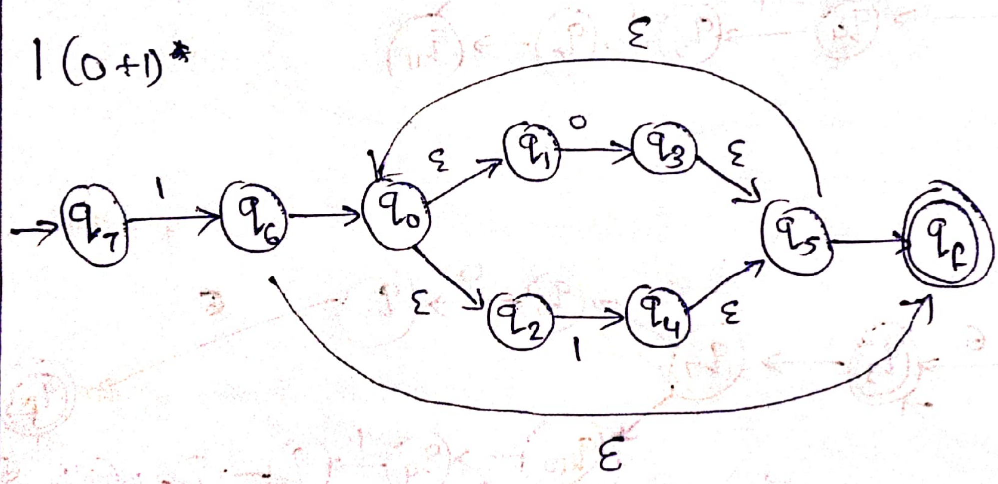
$0+1$



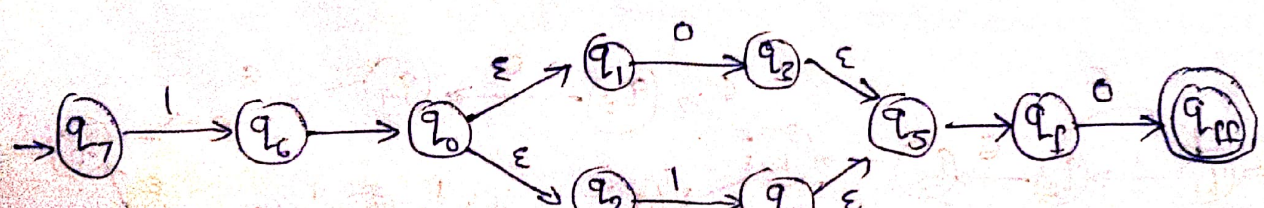
$(0+1)^*$



$1(0+1)^*$

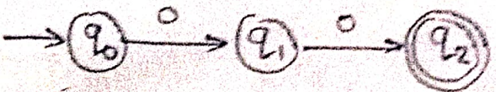


$1(0+1)^*0$

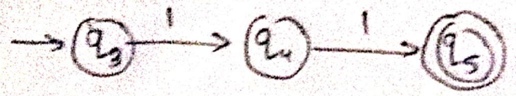


Q (00+11) . 1*

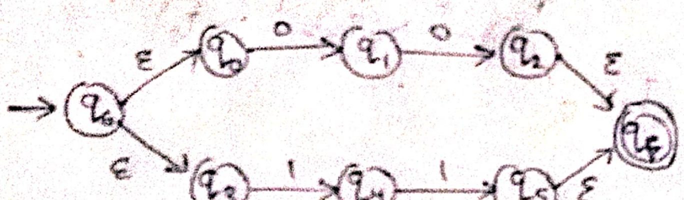
00



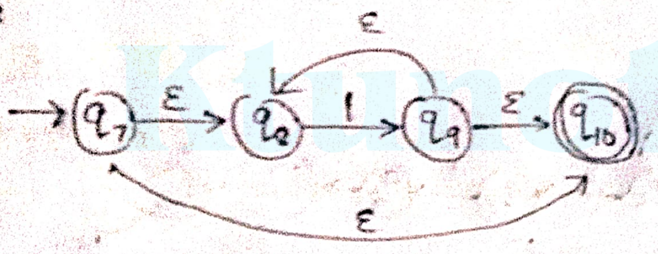
11



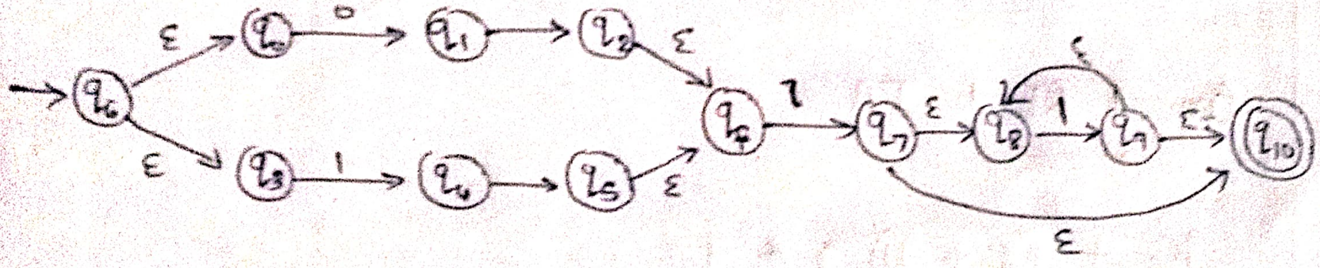
00+11



1*



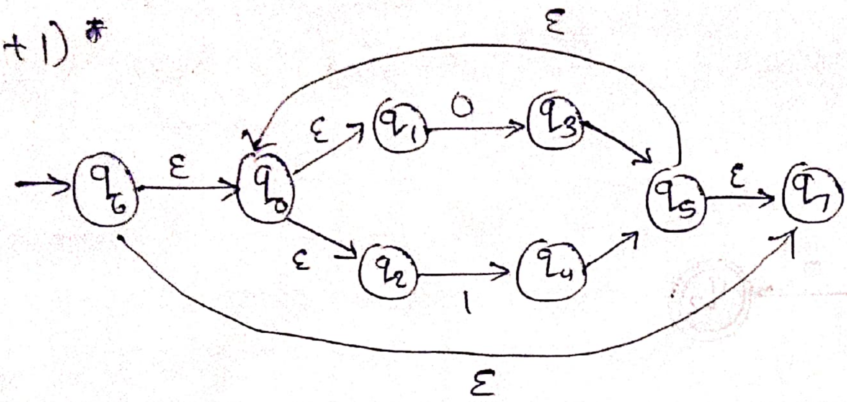
00+11 . 1*



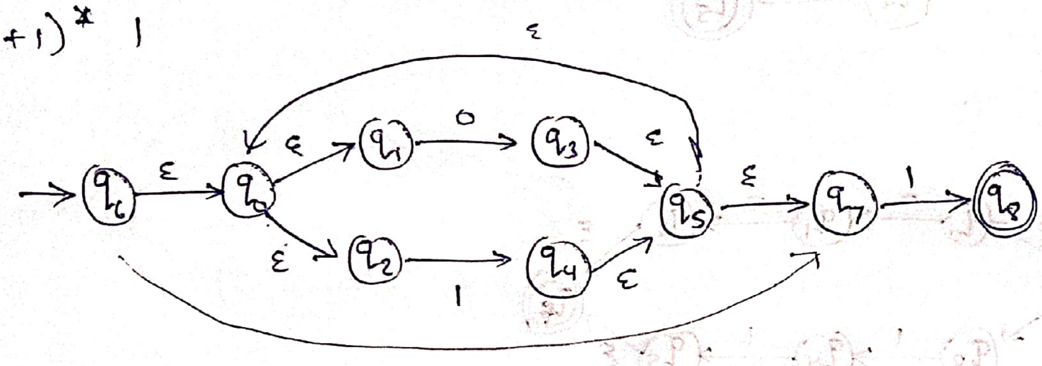
Q (0+1)* 1 (0+1)*

0+1

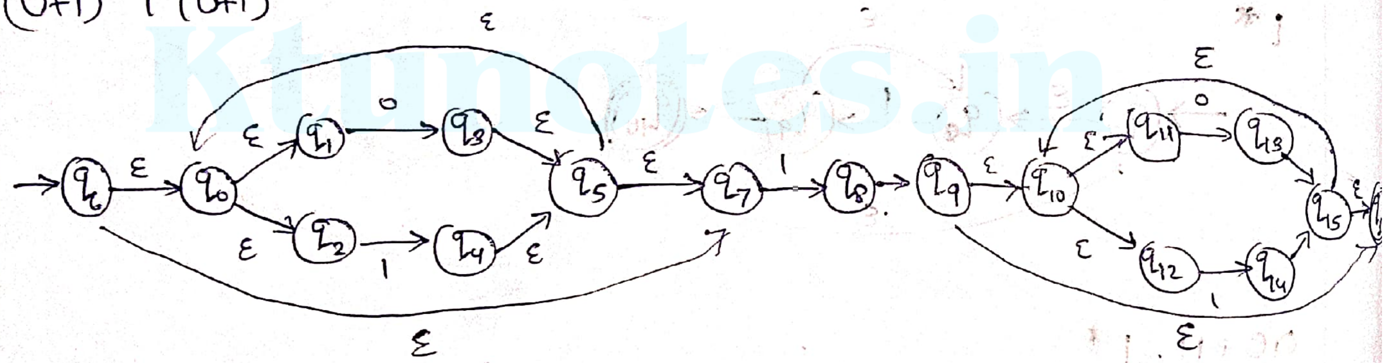
$(0+1)^*$



$(0+1)^* 1$

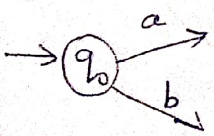


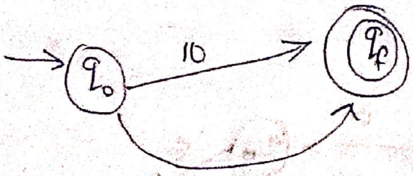
$(0+1)^* 1 (0+1)^*$



Convert RE to FA

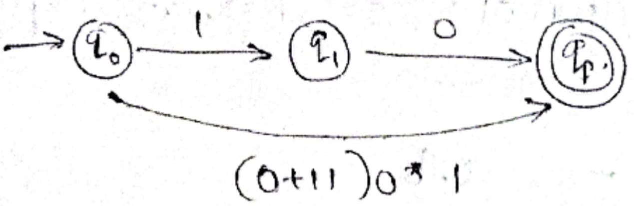
$10 + (0+11)0^*1$

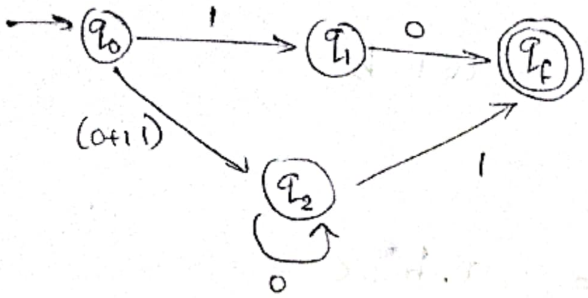


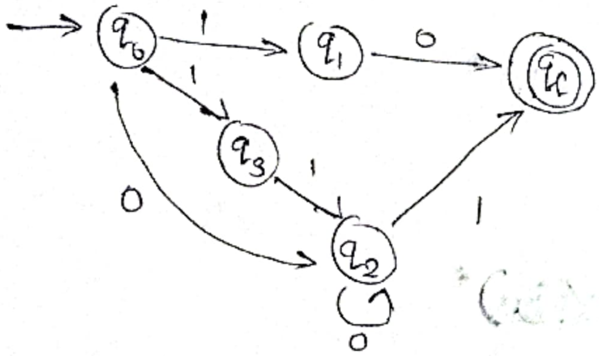


$(0+11)0^*1$

$(0+11) \cdot 0^* \cdot 1$

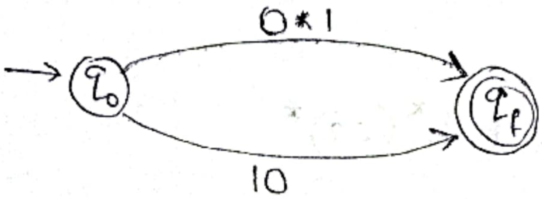


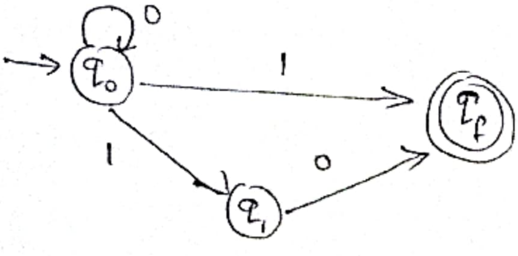




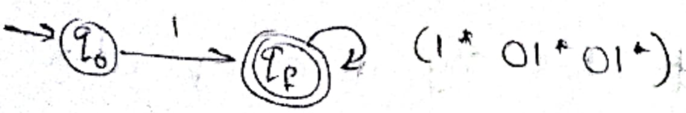
(NFA)

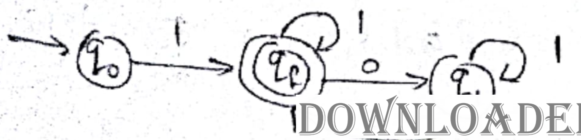
Q $0^*1 + 10$



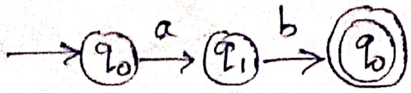


Q $1(1^*01^*01^*)^*$

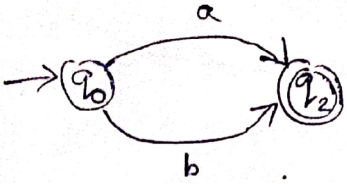




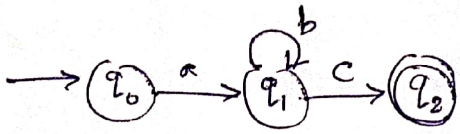
Conversion of Finite Automata to Regular Expression

(1) 

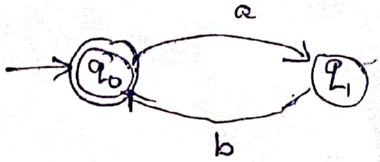
$= a.b$

(2) 

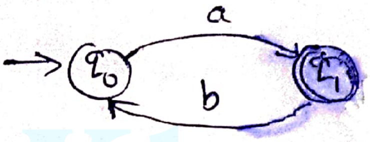
$= a + b$

(3) 

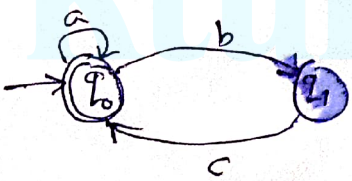
$= a.b^*.c$

(4) 

$= (ab)^*$

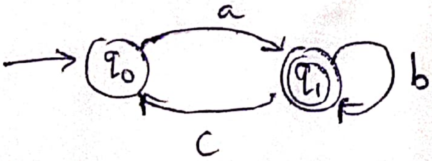
(5) 

$= a(ba)^*$

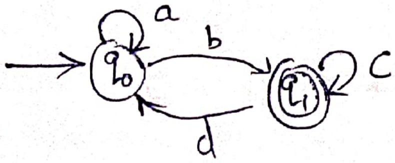
(6) 

= 

$\Rightarrow a^*(bc)^*$

(7) 

$= a.b^*(ca)^*$

(8) 

State Elimination Method

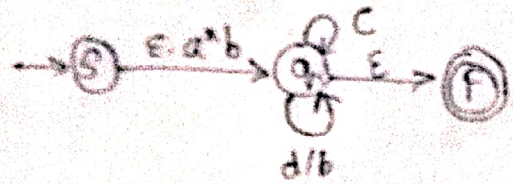
1. If the starting state has any incoming edge create a new initial state and add an ε transition to the

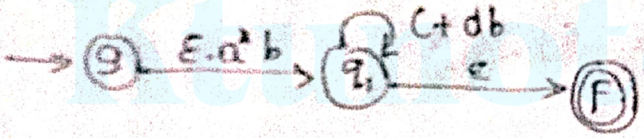
2. If the final state has any outgoing edge create a new final state and add an ϵ transition for the newly created final state to previous final state.

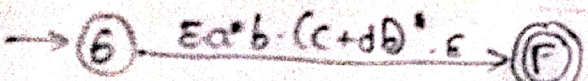
3. Eliminate one by one all states other than initial & final state to get a generalization

8 Ans)

Eliminate q_0

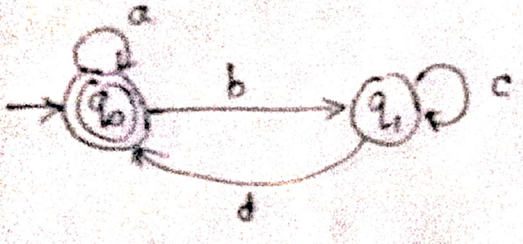




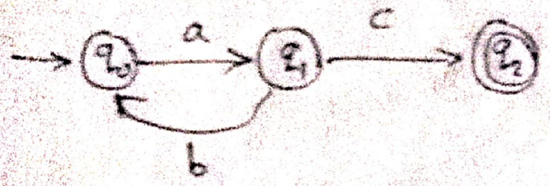


$= a^*b(c+db)^*$

9



10

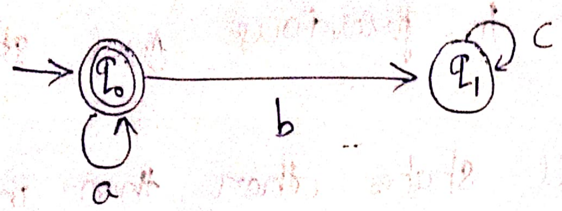


11

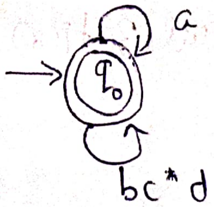
12

Answer -

9)



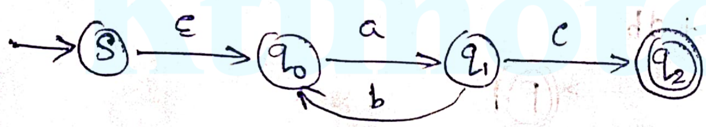
$\Rightarrow$

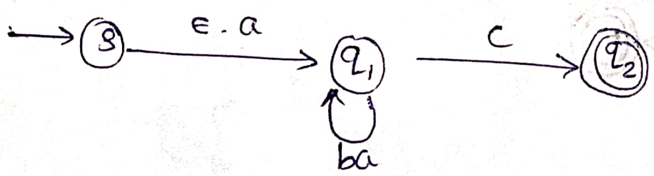


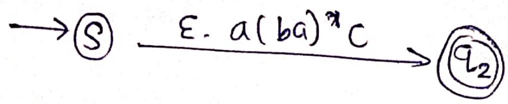
$\Rightarrow (a+bc^*d)^*$

$\Rightarrow \underline{(a+bc^*d)^*}$

10)

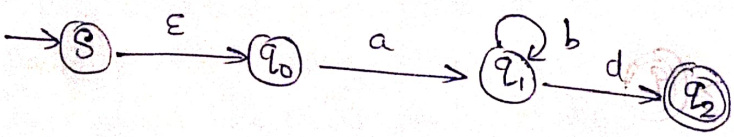


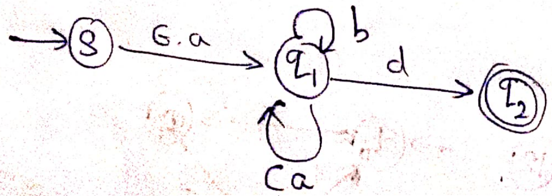


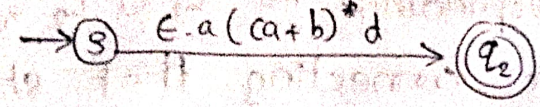


$\Rightarrow a(ba)^*c$

11)

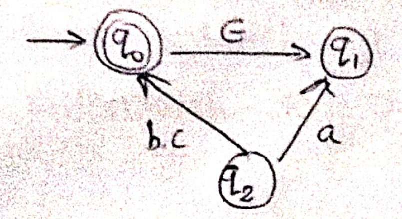




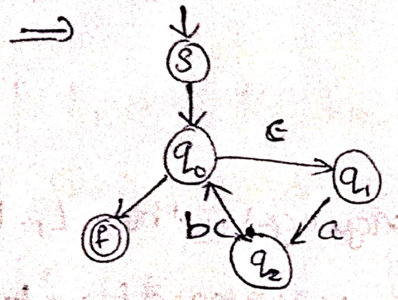


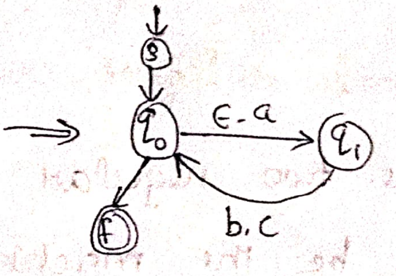
$= a((a+b)^*d$

12)



$\Longrightarrow$





$\Rightarrow (\epsilon.a(b+c))^*$

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

Closure Properties of Regular Language

Set of Regular Language are closed under :

① Union $(L_1 \cup L_2)$

② Concatenation $(L_1 \cdot L_2)$

③ Closure (*) (L^*)

④ Compliment $\bar{L} = \Sigma^* - L$

⑤ Intersection $(L_1 \cap L_2 = \overline{\bar{L_1} \cup \bar{L_2}})$

⑥ Difference $L_1 - L_2 = L_1 \cap \bar{L_2}$

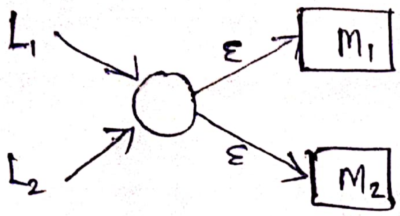
⑦ Reversal L^R

⑧ Homomorphism

1) Union

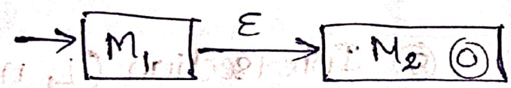
If L_1 and L_2 are two regular languages, then $L_1 \cup L_2$ is regular.

Let M_1 be the automata corresponding to L_1 and M_2 be the a Then you can make an automata corresponding to $L_1 \cup L_2$ by adding a new initial state connecting that state, to the initial state of M_1 and M_2 on ε.



2) Concatenation

If L_1 and L_2 are two regular languages, then $L_1 \cdot L_2$ is also regular. M_1 be the machine corresponding to L_1 and M_2 be corresponding to L_2. Then you can make an automata corresponding to $L_1 \cdot L_2$ by connecting the final state of M_1 to the initial state of M_2 on ε.



3) Closure

Let L be a regular language, then the closure of L (L^*) is also regular. Let R be the regular expression corresponding to L, then R^* is a regular expression corresponding to L^*.

Eg. Let $L = \{a\}$, $L^* = \{\varepsilon, a, aa \ldots\}$

4) Compliment

Let L be a regular language. Then the compliment $\bar{L}$ is also regular.

~~Swa~~ Interchange the final and non-final states in an automata to get its compliment.

5) Intersection

Let L_1 and L_2 be two regular languages, then $L_1 \cap L_2$ is also regular.

$$L_1 \cap L_2 = \overline{\bar{L_1} \cup \bar{L_2}}$$

6) Difference

Let L_1 and L_2 be two regular language, then $L_1 - L_2$ ~~b~~ also is regular.

7) Reversal

Let L be a regular language, then the reverse of L, L_R^R is regular.

Reversal of a FA -:

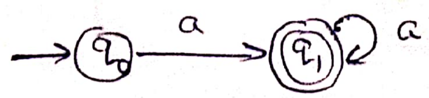
Step 1 - Let M be the FA corresponding to L, and M' be FA corresponding to L^R.

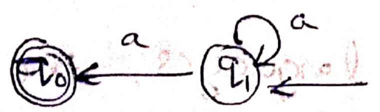
Step 2 - Make the initial state of M as final state of M'.

Step 2 - Make the final state of M, the initial state of M

Step 3 - Reverse the direction of edges of M with M'

Step 4 - No changes in Loop and remove unneccessary states.

M 

M' 

8) Homomorphism

$\Sigma \longrightarrow \Delta^*$

$\rightarrow h(L)$

Eg. $\Sigma = \{0, 1\}$ $\quad \Delta = \{a, b\}$

$h(0) = ab \qquad h(1) = aba$

$h(0100) = h(0) \cdot h(1) \cdot h(0) \cdot h(0)$

$= ab \cdot aba \cdot ab \cdot ab$

Homomorphism is a mapping function ~~from~~ $\Sigma \longrightarrow \Delta^*$, where Σ and Δ are alphabets.

Homomorphic function $h(L)$ for any language L will have a mapping for each of its input alphabet to a string defined in other alphabet.

* Let L be a regular language then the homomorphic function h(L) is closed under the set of regular language.

* Let $0^* 1^*$ correspond to a regular expression for a language L. Let h(0) be ab and h(1) be aba, then the regular expression for h(L) will be $(ab)^* (aba)^*$. Since h(L) is also a regular expression, h(L) corresponds to a regular language

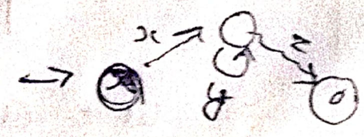
* $h^{-1}(abaab) \Rightarrow$ inverse homomorphism
  $\Rightarrow \{10, 0\}$

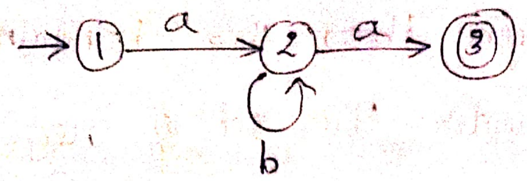
Pumping Lemma for Regular Language

Let 'L' be a regular language (exists a DFA) then there exists a "pumping constant", P (P is the # of states in DFA), for all $w \in L$ with $|w| \geq P$, (w creates a loop in the DFA).

We can write w as xyz such that

1. $|y| \geq 1$ or $|y| \neq 0$
2. $|xy| \leq P$
3. $xy^iz \in L$ for all $i \geq 0$





$w = aba$ $p = 3$

$x\ y\ z$

$aba,\ ab^2a,\ ab^3a,\ ab^4a$

Step 1 - Assume that L is regular

Step 2 - Then there exist a pumping constant p for l

Step 3 - choose one string w in l such that length of w is greater than or equal to p. Therefore

Step 4 - Look at every decomposition of w into xyz such that $|y| \geq 1$ & $|xy| \leq p$

Step 5 - Find one i such that $xy^iz \notin L$

① Prove that the language $\{a^n b^n \mid n \geq 1\}$ is not regular

Ans) 1. Assume the language is regular

2. Let the pumping constant be p

3. w can be divided into 3 parts, x, y, z

s.t $|y| \neq 0$ or $|y| \geq 1$ — ①

$|xy| \leq p$ — ②

$w = a^p b^p$ $z = b^p$

$x = a^m$

$y = a^{p-m}$

$x \cdot \epsilon$ $a^p b^p$ $p < n$ $p \leq s$

$\underbrace{\underbrace{a}_{x}\underbrace{aaaa}_{y}}_{p}\ \underbrace{bbbbb}_{z}$

∵ It is assumed regular.

$w = xy^iz$ ——— ③

$= a^m (a^{p-m})^i b^p$

$= a^m (a^{p-m})^2 b^p$ // i = arbitrary, 2, 3... any no.

$= a^m a^{2(p-m)} b^p$

$= a^m a^{2p-2m} b^p$

$= a^{2p-m} b^p$

$2p - m = p$

$p - m = 0$

$\underline{\underline{p = m}}$

∴ $|y| \neq 0$ ~~or~~ $|y| \geq 1$ is violated! (y can't be 0)

∴ Assumption is wrong

∴ L is not regular.

Q | Prove that the language $\{ww \mid w \in \{0,1\}^*\}$ is not regular

Ans. 1. Assume the language is regular

2. Let the pumping constant be 'p'

3. Choose a string

$w = 0^p 1$ $\qquad |y| \neq \varepsilon$

$ww = 0^p 1 0^p 1$ $\qquad |xy| \leq p$

$x = 0^{\cancel{k-x}}$

$y = 0^k$

$z = 0^{p-(\ldots)}$

$xy^iz \in L$

$xy^iz = 0^l(0^k)^i 0^{p-(k+l)} 1 0^p 1$

$= 0^l 0^{2k} 0^{p-k-l} 1 0^p 1$

$= 0^{k+p} 1 0^p 1$

$k+p = p$

$k = 0$

$w = \underbrace{000011000011}_{x\ y\ z}$

$p = 4 \quad |xy| \leq p$

$= 00(0)^2 011000011$

$= 0000011000011 \notin L$

(y)

$\therefore$ Assumption is wrong

$\therefore$ Language is not regular.

Ultimate Periodicity

$U = \{3, 7, 11, 20, 23, 26, 29, \ldots\}$

p is called period.

$U \subseteq N$ is said to be ultimately periodic if

$\exists_n \geq 0$ & $p > 0$ $\forall m \geq n$, $m \in U$ iff $m+p \in U$

If we take an $m \in U$ then $m+p \notin U$ or $m+p \in U$

$\Rightarrow m \in U$

→ consider $U = \{3, 7, 11, 20, 23, 26 \ldots\}$

choose $n = 20$ & $p = 3$

then $\forall\ m \geq 20$, $m + p$ or $m + 3 \in U$

$m > n$

$m + p \in U$

$L = \{a, aa, aaa, \ldots\}$

$U = \{1, 2, 3, \ldots\}$