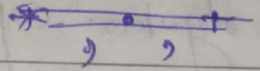
* Regular Expression

Regular Operator :-

$*$, $\cdot$, $+$ → Union

Klean Closure — Concatination

→ Precedency → $*$, $\cdot$, $+$

R.E 

R.E	R.L
$r = \phi$	$L = \{\phi\}$
$r = \epsilon$	$L = \{\epsilon\}$
$r = a$	$L = \{a\}$
$r = a + b$	$L = \{a, b\}$
$r = a \cdot b$	$L = \{ab\}$
$r = a^+$	$L = \{a, aa, aaa \ldots\}$
$r = a^*$	$L = \{\epsilon, a, aa, aaa, \ldots\}$
$r = (a+b)^+$	$L = \{a, b, aa, ab, ba, bb \ldots\}$
$r = w_1 + w_2 + w_3 + \ldots + w_n$	$L = \{w_1, w_2, \ldots, w_n\}$
$r = (a+b)^*$	$L = \{\epsilon, a, b, ab, \ldots\}$

Regular Expression :- If Σ is any alphabet then an expression which is constructed over the alphabet Σ using the operator $*$, $\cdot$ and $+$ is called regular expression.

OR

An expression that represents regular language is called as regular expression.

- If r is regular expression then $L(r)$ is language generated by r.
- Regular expression always represents regular language.
- Every finite language is regular.
- If r is regular expression then both r^* and r^+ are also regular expressions.

*Q] Construct the regular expression that generates all strings of 0s and 1s including $\in$ and excluding $\in$.

→ 1] $r = (0+1)^*$

2) $r = (0+1)^+$

Q a) start with 10

b) end with 10 c) Contains substring 10

→ 1] $L = 10X$

$r = 10(0+1)^*$

2] $r = (0+1)^* 10$

3) $r = (0+1)^* 10 (1+0)^*$

Q) 4) Start & end with 0 5) start & end with same, ~~diff~~ symbol

c) ~~——~~ diff^n symbol.

4] 5] $r = 0(1+0)^*0 + 0$

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

7] Start with 00 or 11 8) End with 00 or 11

9) Contain ~~sorbs~~ substring 00 or 11

10) Start and end with 00 or 11.

7] $r = 00(1+0)^* + 11(1+0)^*$

8) $r = (1+0)^*00 + (1+0)^*11 = (00+11)(1+0)^*$

9) $r = (1+0)^*00(1+0)^* + (1+0)^*11(1+0)^*$

10) $r = 00(1+0)^*00 + 00 + 11(1+0)^*11 + 11$

$(00+11)\cdot(1+0)^*\cdot(00+11) + 00 + 11$

=

9) Construct reg exp that contain all strings of 0's and 1's where

1] Third symbol from left hand is 0.

2) Fourth symbol from right is 0

→

1) $r = (0+1)\cdot(0+1)\cdot 0\cdot(0+1)^* = (0+1)^2\cdot 0\cdot(0+1)^*$

2) $r = (0+1)^*\cdot 0\cdot(0+1)\cdot(0+1)\cdot(0+1)$

$= (0+1)^*\cdot 0\cdot(0+1)^3$

9) Construct regular expression that contains all strings of 0's and 1's, where no. of zeros is

1) exactly 2 2) at most 2 3) at least 2

4) divisible by 3.

→ $L = 0\times 0 + 00\,11 + 11\,00$

$r = 01^*0 + 00\,1^* + 1^*00$

$= 1^*0\cdot 1^*0\,1^*$

2) $r = 1^*01^*01^* + 1^*01^* + 1^*$

$=$

3) ~~$r = 1^*01^*01^*01^*$~~

$r = 1^*01^*0(0+1)^*$

4 ~~3~~ $=$

4 ~~3~~] $r = 1^*(1^*01^*01^*01^*)^*1^*$

Q] Construct regular express that generate all strings of 0s and 1s where length of string is

i) exactly 2 ii) at most 2 iii) at least 2

iv) even v) odd vi) 2 mod 3.

→ 1) $r = (0+1)\cdot(0+1) = (0+1)^2$

2) $r = (0+1)^2 + (0+1) + \epsilon$

3) $r = (0+1)^2(0+1)^*$

4) $r = [(0+1)^2]^*$

5] $r = $ ~~$(0+1)^*$~~ $[(0+1)^2]^*(0+1)$

6] $|w| = 2 \bmod 3$

~~$r = [(0+1)^2]^*$~~

$r = (0+1)^2[(0+1)^3]^*$

Q] If string starts with 0 then length of string is even, if starts with 1 then length is odd.

→ $r = 0\cdot[(0+1)\cdot[(0+1)^2]^* + 1\cdot[(0+1)^2]^*$

Q] Every string starts with 0 and do not contain 2 consecutive one.

→ ~~$r = 0 \cdot [(10)^* + (01)^* + 10)^*]$~~

$$r = 0[10 + 01 + 0]^*$$
$$= [0 + 01]^+$$

Q] Construct regular expression that generates all strings of 0's and 1's do not contain two consecutive zeores or two consecutive ones.

→ $r = [01)^* + (10)^* + 0 + 1$

→ does not contains 010

$$r = (01)^* 0 + (10)^* 1 + (01)^* + (10)^* + \epsilon$$
$$= (01)^* (0 + \epsilon) + (10)^* (1 + \epsilon)$$

or

$$r = (0+\epsilon)(10)^*(1+\epsilon) \text{ or}$$
$$r = (1+\epsilon)(01)^*(0+\epsilon)$$

Q] i) $L = \{a^m b^n \mid m+n = \text{even}\}$
ii) $L = \{a^m b^n \mid m+n = \text{odd}\}$

* Equivalence between FA and R.E :-

1] FA → R.E

2 ways

(a) ARDEN'S LEMMA

(b) State Elimination method

* ARDEN'S LEMMA :-

This mechanism is used only for DFA and NFA and can't be used for ε NFA.

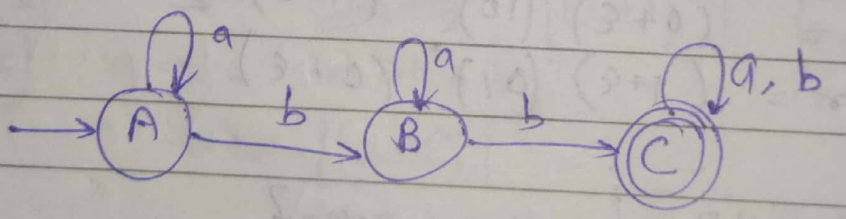
Let p, q, r be three regular expression such that $r = q + rp$ then

1] If p is free from ε then equation has unique solution and solution is

$$r = qp^*$$

2] If p contains ε then eqn $r = q + rp$ has infinitely many solutions.

Q]



$$A = \varepsilon + A \cdot a$$
$$R \quad Q \quad R \quad P$$

P is free from ε

$$\therefore A = \varepsilon a^*$$

$R = Q + RP$

$B = A \cdot b + Ba$

P is free from ε

$B = A \cdot ba^*$

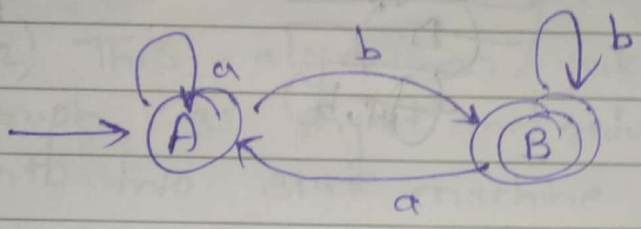
$B = a^* b a^*$

$C = \underset{Q}{Bb} + \underset{R}{C \cdot} \underset{P}{(a+b)}$

$C = Bb(a+b)^*$

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

9]



~~$A = \varepsilon B + A \cdot a$~~

~~$= B \cdot a a^*$~~

$A = (\varepsilon + B \cdot a) + A \cdot a$ (R Q P)

$= (\varepsilon + B \cdot a) a^*$

$= A \cdot a + (A \cdot b^+) a + \varepsilon$

$= A(a + b^+ a) + \varepsilon$

$B = A \cdot b + B \cdot b$

$= A \cdot b b^*$

$= A \cdot b^+$

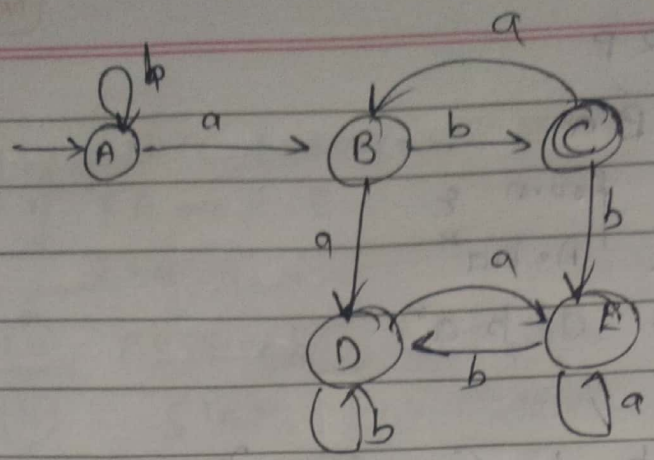
$= (\varepsilon + B \cdot a) a^* b^+$

$R = q p^*$

$= \varepsilon (A a + b b^* a)^*$

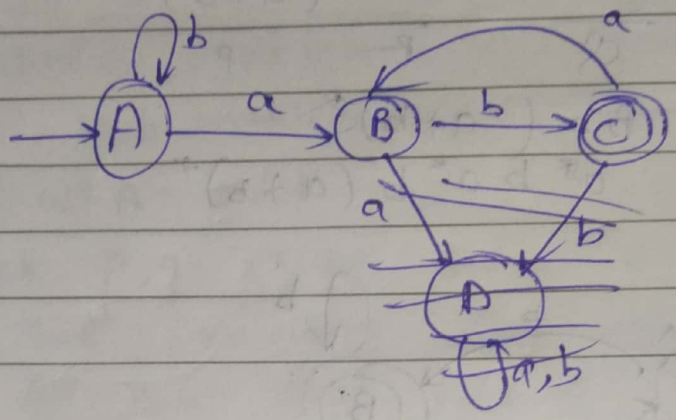
$= (a + b b^* a)^* b b^*$

Q]



→

⇒



Eliminate dead stat equaled stat

$R_A = Q + RP$

$A = \varepsilon + Ab$

$\therefore A = \varepsilon b^*$

$B = B \cdot a + B \cdot C \cdot a + B \cdot \varepsilon$

$B = (B \cdot a + C \cdot a) \cdot$

* Transition Graph :-

The NFA which has more than one initial state and edge level from Σ^* is called as Transition Graph.

It can be represented by 5 tuples.

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

I → set of all initial states.

* State Elimination Method :-

1) This mechanism is not only used for FA but also used for transition graph.

2) This algorithm takes transition graph as input and reduce. transition graph into two state machine with one is initial state and second is final state.

* Algorithm :-

1] Simplify the transition graph to have one initial state and one final state.

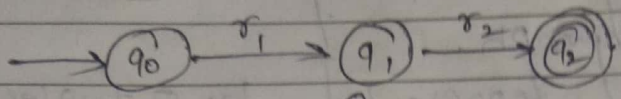
2] Simplify the transition graph to have different initial & final states.

3] If there exists more than one edge between pair of state in same direction then they are called as parallel edge and combine them into a single edge using + operator.

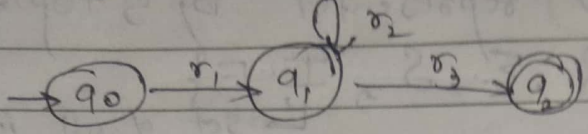
4]

Eliminate state q_1 from the following graph.

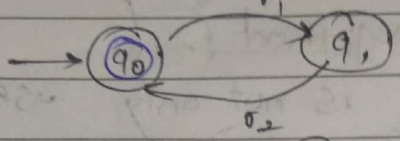
1)



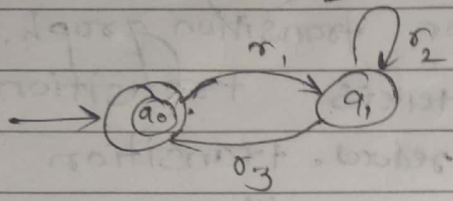
2)



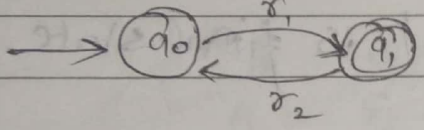
3)



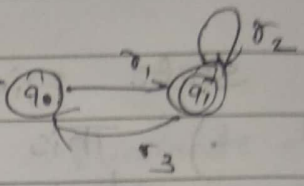
4)



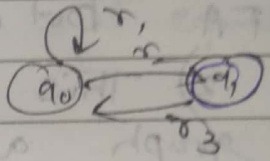
5]



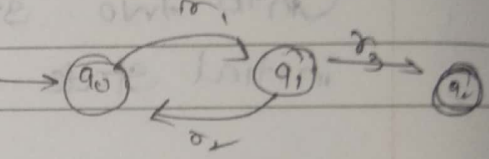
6]



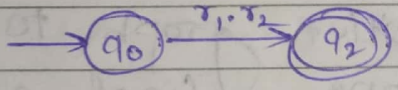
7]



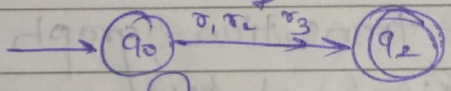
8]



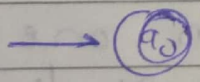
1] ⇒



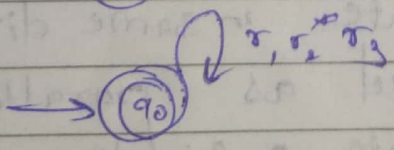
2] ⇒



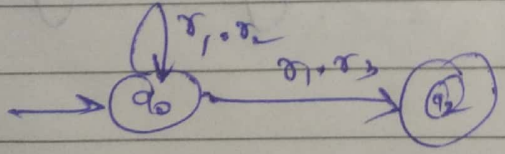
3 ⇒



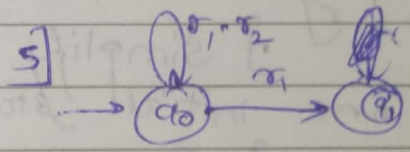
4) ⇒



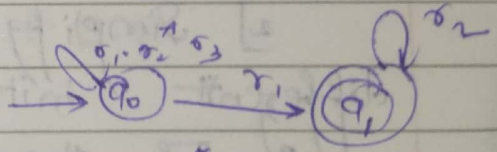
8]



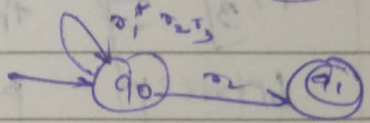
5]



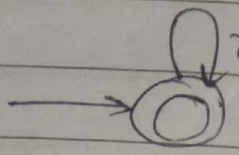
6]



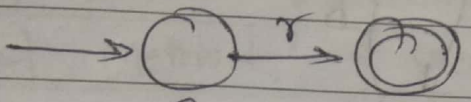
7]



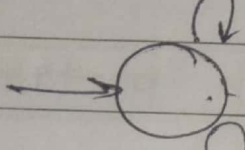
4) Continue the process of state elimination till the transition graph takes one of the follow form.

1) 

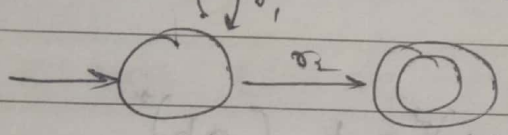
→ R.E = r^*

2) 

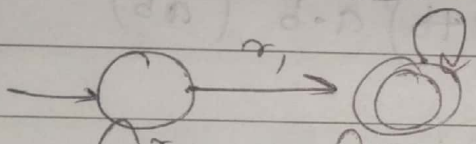
= r

3) 

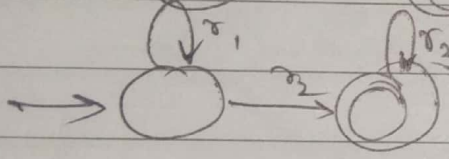
$= (r_1 + r_2)^*$

4] 

$= r_1^* \cdot r_2$

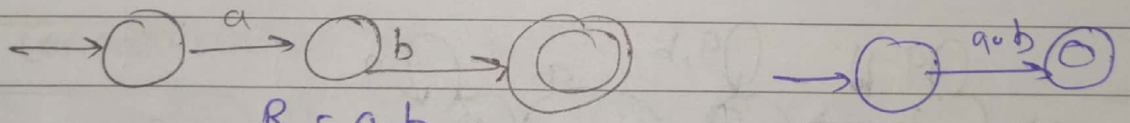
5] 

$= r_1 \cdot r_2^*$

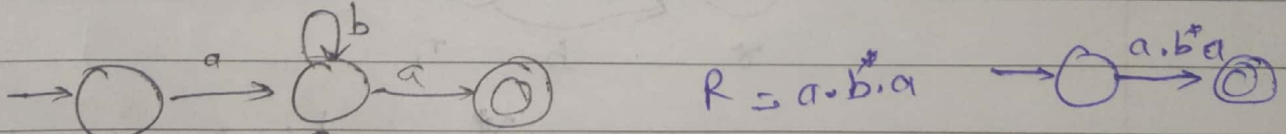
6] 

$= r_1^* r_2 r_3^*$

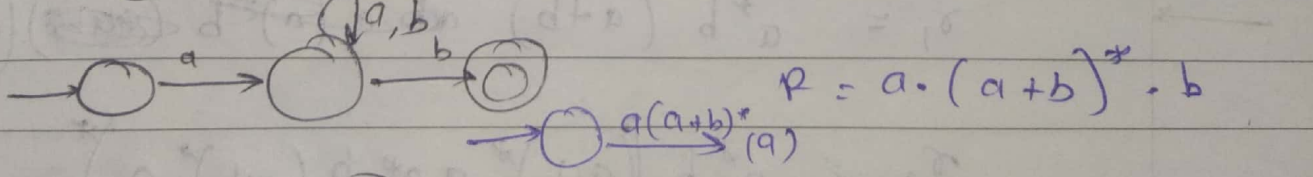
Ex.

1] 

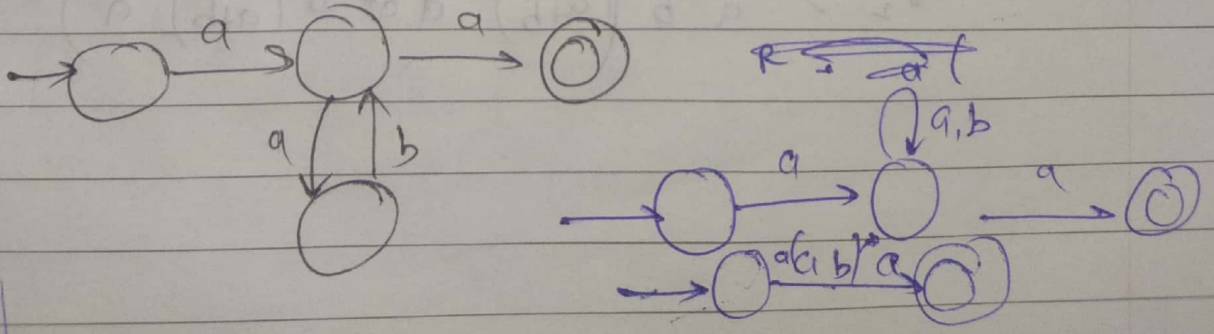
R = a.b

2] 

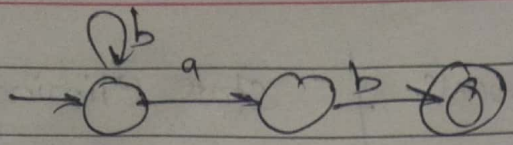
R = a.b*.a

3] 

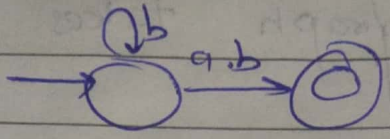
$R = a \cdot (a+b)^* \cdot b$

4] 

5]

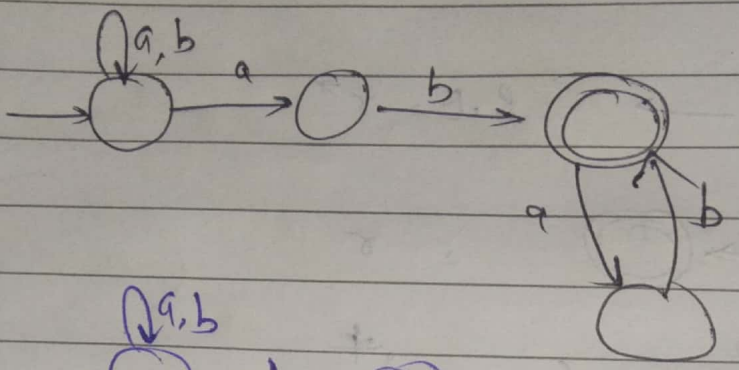


→

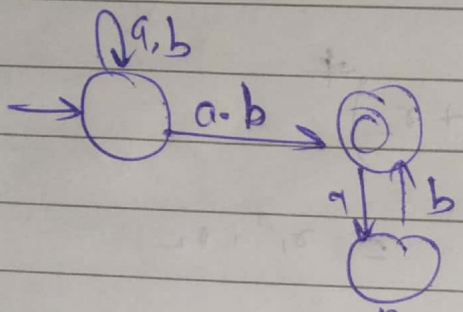


$R = b^* ab$

6]

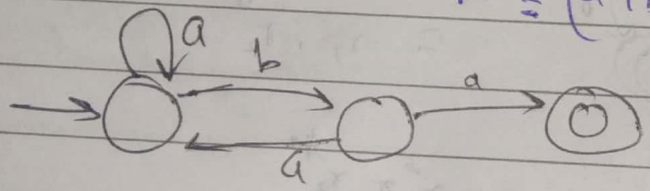


→



$R = (a+b)^* a.b\ (ab)^*$

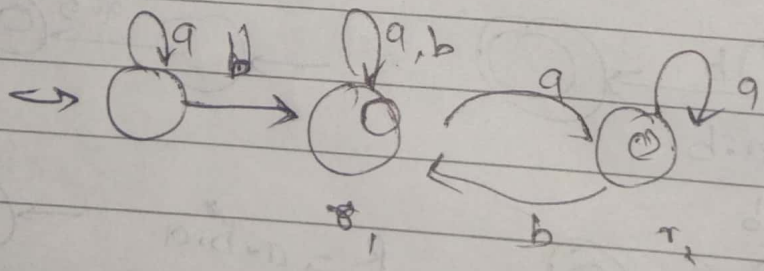
7]



→

$R = a^* (ba)^* a$

8)

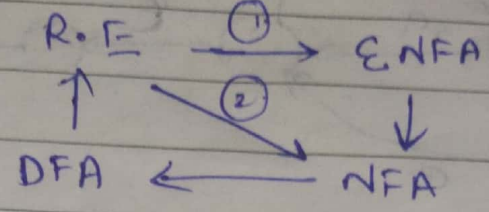


→

$r_1 = a^* b\ (a+b)^*$ ~~$a^* (a)^* b\ (ab)\ (a\, a^* b)^*$~~

$r_2 = a^* b \left((a+b)^* a\, a^* b\, (a+b)^* a \right)^*$

Equivalence between R.E and F.A:-

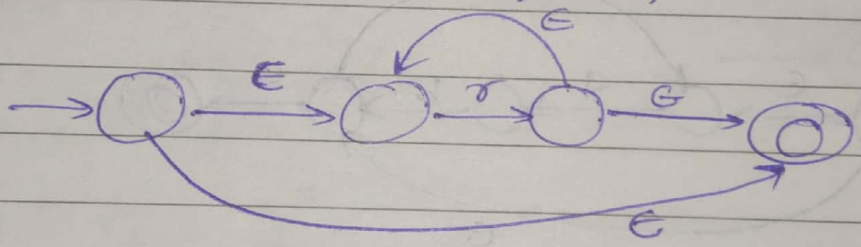


1] Method of Synthesis
2] Method of Decomposition.

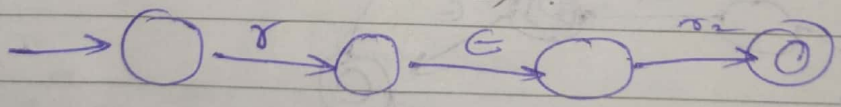
* Method of synthesis :-

1] Kleen. Closure (r^*)

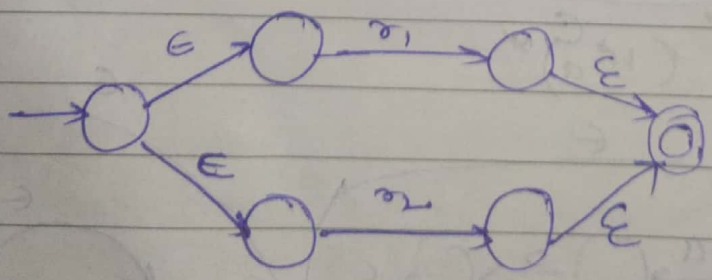
$r^* = \epsilon, r, rr, rrr, \ldots$



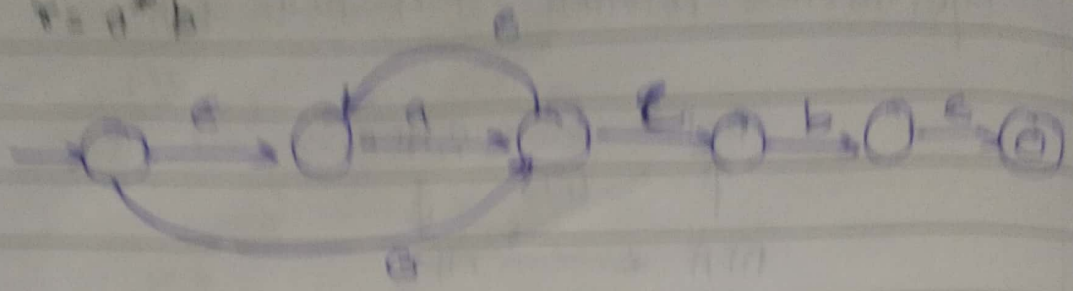
2] Concatenation (r_1, r_2) :-



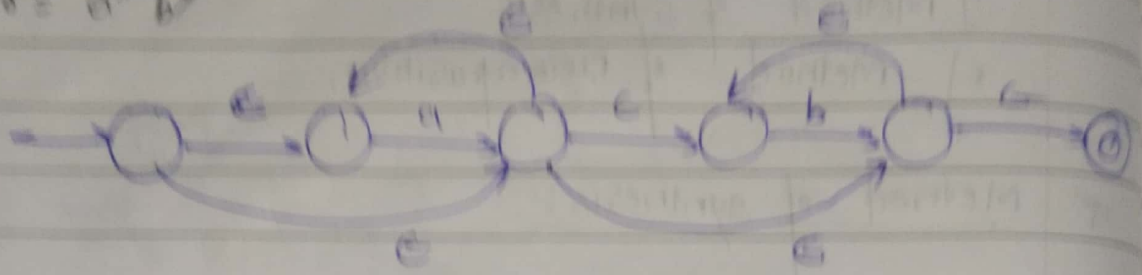
3] Union ($r_1 + r_2$) :-



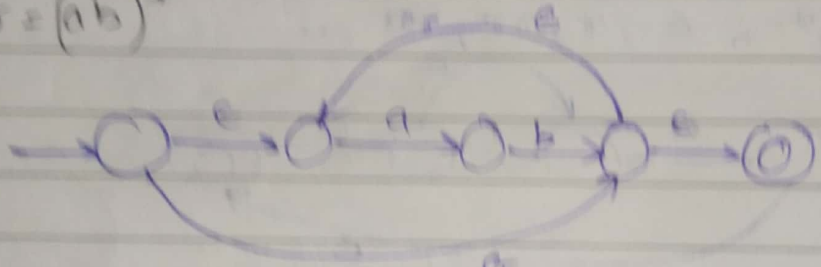
i) $r = a^* b$



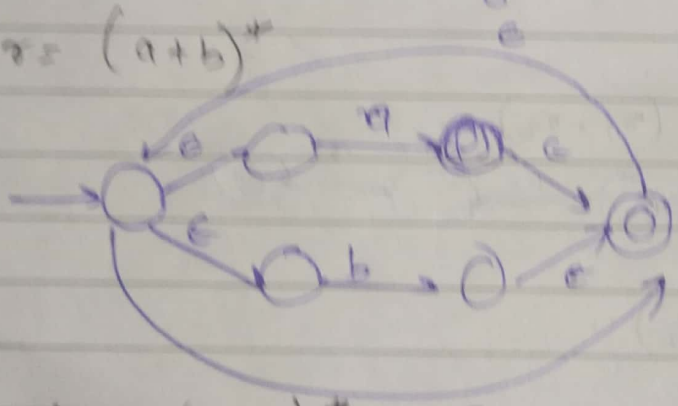
ii) $r = a^* b^*$



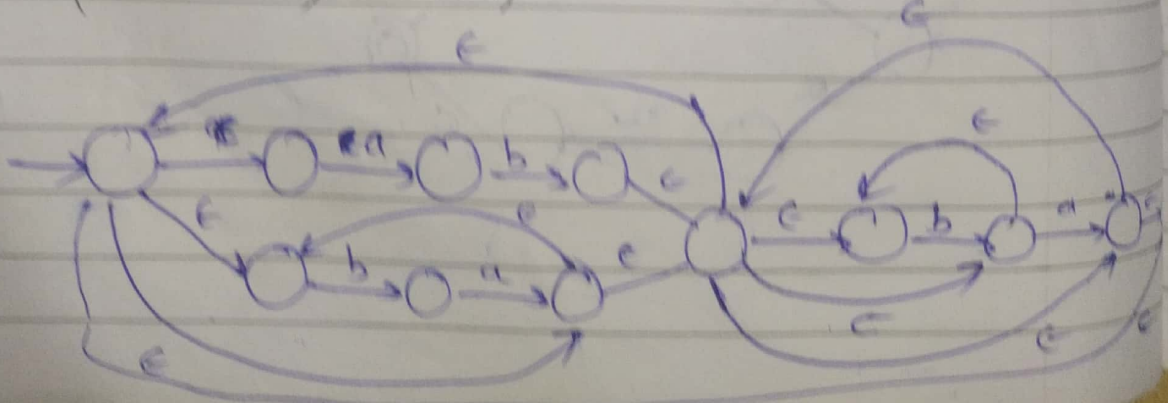
3) $r = (ab)^*$



4) $r = (a+b)^*$



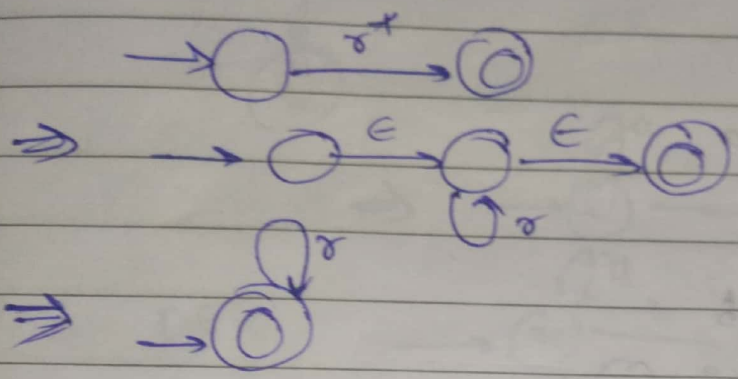
5) $r = (ab + (ba)^*)^* (b^* \frac{c}{a})^*$



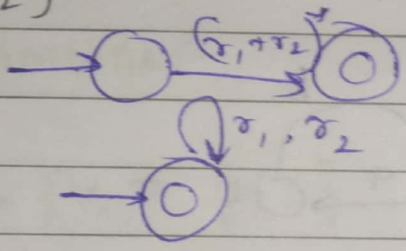
8086 is a 16 bit processor

Method of Decomposition (State creation Method) :-

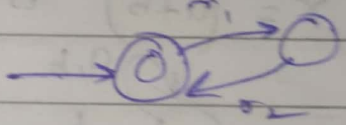
r^*



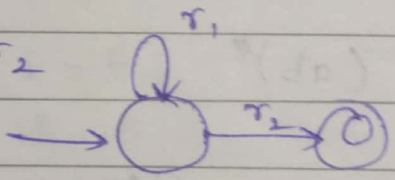
2) $(r_1 + r_2)^*$



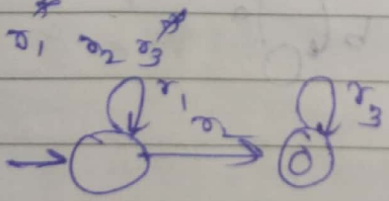
3) $(r_1 \cdot r_2)^*$



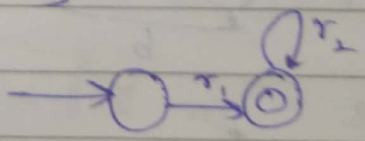
4) $r_1^* \; r_2$



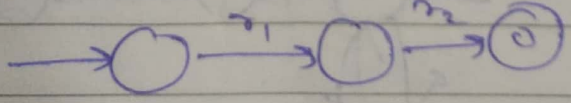
5) $r_1^* \; r_2 \; r_3^*$



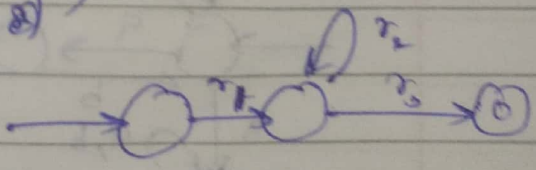
6) $r_1 \; r_2^*$



$r_1 \cdot r_2$



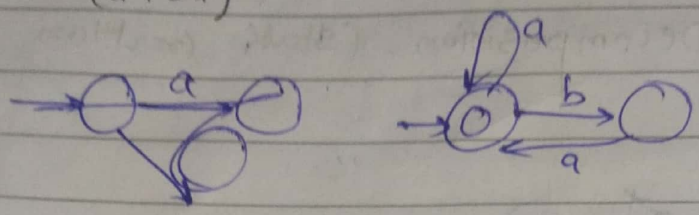
8) $r_1 r_2^* r_3$



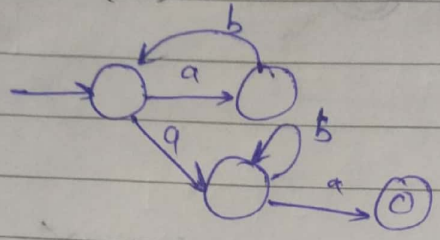
Ex

1] $r = (a+ba)^*$

→

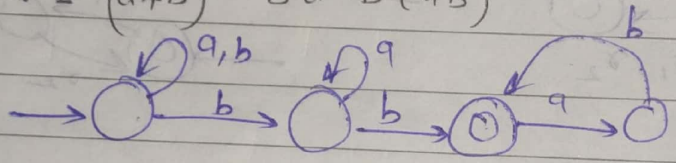


2) $r = (ab)^* ab^* a$



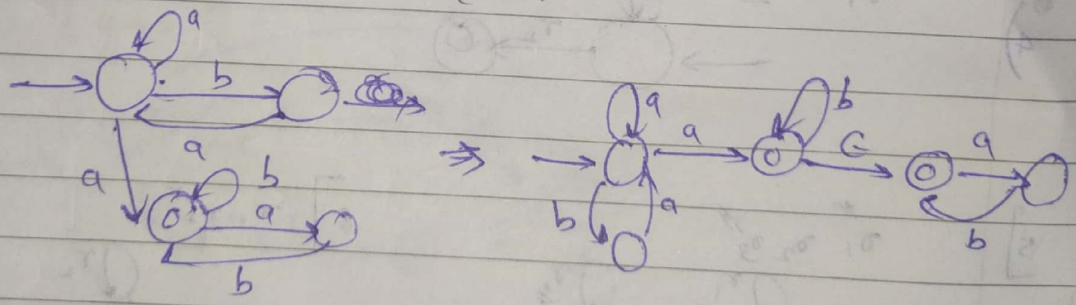
3) $r = (a+b)^* ba^* b(ab)^*$

→



4) $r = (a+ba)^* ab^* (ab)^*$

→



5] $r = (ab)^* b^* a (a+b)^*$

→