Date: ________ Day: M T W T F S

TOA - 1

pre-req of compiler construction.

Automata → model / machine

Language ⇒ Medium of communication.

Formal Language	**Informal Language**
⇒ set of rules defined	• verbal + gesture meanings.
⇒ well defined protocol e.g c++.	• d/f meanings on same verbal if d/f gestures are used.
⇒ only verbal meanings, same meaning each time	

Building blocks of language

(Informal) **Eng** ⇒ Alphabets (26 char) ⇒ formation of letters (dictionary)

⇒ sentence (grammar rules).

(formal) **C++** ⇒ ASCII (256 char).

⇒ words (rules) (Infinite possible).

⇒ Instructions.

• for checking valid variable names ⇒ construct rules (automata)

Date: Day: M T W T F S

TOA-2 (16/01/23)

- valid & invalid string.
- D/f Language.
- Σ/set of alphabets for language.

Represent a Language

1) Descriptive.

$$\Sigma = \{0,1,\ldots,9\}$$

Language of odd no :- $\{1,3,5,\ldots\}$

can be written in set builder form.

2) Regular Expression.

- Some language are regular/some not.

language that can be represented in regular form.

3) Context free Grammar.

Descriptive form

$\Sigma = \{0,1\}$ → can form infinite strings.

$= \{0,1,00,01,000,\ldots\}$

Date: Day: M T W T F S

- Language start with 0 → L1 → {0, 01, 001, 011, 000, ...}
  can be subset of other lang.

  word ⇒ valid string belonging to a language.
  every word is string. but not every string is word.

- Language start & end with d/f char → L2 ⇒ {01, 10, 001, ...}

- Language which contains equal no of 0's & 1's → L3
  ⇒ {01, 10, 1100, ...}.

- Language with equal 0's & 1's & 0's are formed before 1's → L4 ⇒ {01, 0011, 000111, ...}.
  ⇒ $\{0^n 1^n, n >= 1\}$.

- Palindrome Language ⇒ L5 ⇒ {0, 1, 00, 010, ...}.
- Palindrome language with length < 5 ⇒ finit set.

Regular Expression.

- Regular exp can be difficult.

→ Password Validation

$$\Sigma = \{0, 1, \$, \#, A, \ldots, F, a, \ldots, f\}$$

- atleast 1 special char.
- atleast 1 digit.
- atleast 1 capital char
- min length 5

Date: Day: M T W T F S Date:

$\Rightarrow \{A\$001\}$ valid.

$\{a\$001\}$ invalid.

- Regular exp have some limitations.

Operators:

1) Kleen Star $\star$

Repeat 0 or more/infinite time.

2) Kleen Plus $+$

Repeat 1 or more time.

Examples: $\Sigma_1 = \{a\}$

$\Sigma_2 = \{0,1\}$

$\Sigma_1^{\star} = a^0 = \lambda$

$= a^2 = aa$.

$\Sigma_1^{\star} = \{\lambda, a, aa, \ldots\}$.

$\Sigma_2^{\star} \Rightarrow (0,1)^{\star} \Rightarrow$ Kleen star on whole set.

$\Rightarrow (0,1)^0, (0,1)^1, (0,1)^2, \ldots$

each possible combination. $\lambda,\ 0,1\ \ 00,01,10,11, \ldots$

Kleen Plus $= \Sigma_1^{+} = (a)^{+} \Rightarrow a^1, a^2, \ldots$

(null/empty not included). $a, aa, aaa, \ldots$

Day: M T W T F S

Date:

String Operations:

- length of string.. $\Rightarrow |aaa| = 3$ (cardinality operator)
- reverse string.
- concatenate string.

↳ $s_1 = a$, $s_2 = ab$.

$s_1 . s_2 = aab$.

$s_1 = \lambda$, $s_2 = ab$.

$s_1 . s_2 = \lambda . ab = ab$.

Recursive Def

↳ self repeating until base condition.

$\Sigma = \{0, \ldots, 9\}$.

Language of integers $= \{0, \pm 1, \pm 2, \ldots\}$.

1) Some strings are the part of the language.
2) Define some rule that generate the new strings of the language.
3) Any string which produce from rule 2 belongs to language other than that no string would be part of language.

Date: Day: M T W T F S

Example:

Rules for creating int language:

$\Sigma = \{0, \pm 1, \pm 2$ (→ base string/condition)

$x \in \Sigma \Rightarrow x+1$ or $x-1 \in \Sigma$

Rules for creating even no language

$\Sigma = \{2, 4, 6, \ldots\}$.

$x \in \Sigma \Rightarrow x+2$ or $x-2 \in \Sigma$

Polynomial

$5x^0, 6x^3, 2x+3x^2$.

$\Sigma = \{$ Any number, $x \}$.

If p & q are polynomial.

Then $p+q$, $p-q$, $p \times q$ are also polynomial.

$$\underbrace{2x}_{p} + \underbrace{\underbrace{3x}_{p}\ \underbrace{x}_{q}}_{q}$$

TOA-3 (22/01/23)

- **2nd Representation - Regular Expression.**

Operators:-

1) Kleen Star $(*)$ — 0 or more rep
2) Kleen Plus $(+)$ — 1 or more rep.

Date: Day: M T W T F S

3) Concatenation ($.$) $(a)(b) = ab$ $a.b = ab$

4) Option/Choice ($|$, $+$) $(a+b|c)$

"either a or b or c.

- **Represent a Language:**

$\Sigma = \{a, b\}$.

Multiple RE are possible

1) Language which produce strings that only contain as.

$\Rightarrow \{a, aa, aaa, \ldots\}$ Descriptive form.

R.E $\Rightarrow a^+$

2) Language which produce strings that contains a's or b's.

D.F $\Rightarrow \{a, b, ab, ba, \ldots\}$.

R.E $\Rightarrow (a+b)^+$ or $(a|b)^+$

Check: 1) $(a+b)^* \Rightarrow \lambda, a, b, \ldots$

2) $a^*.b^* \Rightarrow \{\lambda, a, b, ab, aa, bb, \ldots\}$.

↳ due to concatenation sequence matters & a cannot be after b.

3) $a^* + b^*$

↳ $\{\lambda, a, aa, aaa, \ldots, b, bb, bbb, \ldots\}$.

Date: Day: M T W T F S

because choice is after * operator
(producing combination).

$(a+b)^2 = (a+b)(a+b)$.
$= ab, aa, ba, bb$.

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

$\Rightarrow \{\lambda, a, b, ab, ba, aa, \ldots\}$

- to produce ab from this $\Rightarrow (a^* + b^*)^2$
$= (a^* + b^*)(a^* + b^*)$
$= (a^1 + b^0)(a^0 + b^1)$
$= a.b = ab$.

3) Language that starts with a.

$\Rightarrow \{a, ab, aab, \ldots\}$

$a.(a+b)^*$ ✓

$a.(a^* + b^*)^+$ ✓

$(a^+ . b^*)^+$ ✓

$(a^+ + b^*)^+$ ✗ ba also possible

TOA-4 (23/01/24)

Regular Expression:

$\Sigma = \{0, 1\}$.

1) Start & end with same letter.

$\{0, 1, 00, 11, 101, 111, \ldots\}$

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

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

$\boxed{[A\text{-}Z] \Rightarrow \text{range} \rightarrow \text{class Operator}}$

2) Start & end with d/f letter.

$\{01, 10, \ldots\}$.

R.E ⇒ $0.(0+1)^*.1 + 1.(0+1)^*.0$.

3) Language which contains atleast 2 0's.

$\{00, 100, 010, 001, 1000, \ldots\}$.

R.E ⇒ $(0+1)^*.0.(0+1)^*.0.(0+1)^*$.

4) Language which contain consecutive 2 zero's.

R.E ⇒ $(0+1)^*.00.(0+1)^*$

5) Even no of 0's. $E = [0, 2, 4, \ldots]$.

$\{1, 11, 111, \ldots, 00, 001, 010, 101, \ldots\}$

$(1^*.00.1^*)^*$ X

R.E ⇒ $(1^*.0.1^*.0.1^*)^* + 1^*$

Date: Day: M T W T F S

6) Odd no of 0's.

$(1^*.00.1^*)^*.0.(1^*.00.1^*)^*$

$\{0, 01, 10, 0100, \ldots\}$.

7) Even no of 0's & even no of 1's.

$\{00, 0000, \ldots, 11, 1111, \ldots, 0011, 0110, 1100, 1001, \ldots\}$.

R.E $= (11^*.0.1^*)^*$ ✗

$= ((00+11)^* + (01+10)(00+11)^*(10+01))^*$

1000 0001

$\Sigma = \{$Alphabets, Digits, _$\}$.

8) start with _ or Alphabet.

1) (Alphabet + _).(Alphabet + Digit + _)*

2) ((Alphabet + _)* . Digit*)*

int a = 4501;

↳ check is it valid num/int ?
compiler has pattern stored not all the numbers.

↳ patterns have reg exp behind them.

↳ Reg expression for comment also.

TOA-5 (29/04/24)

Construct R.E for a lang:

1) that doesn't contain substring ab.

deny pattern is bit difficult

$\{\lambda, a, b, ba, aa, bb, aaa, bbb, bba, \ldots\}$.

R.E $= a^* + b^*.a^* \Rightarrow b^*.a^*$

as multiple can be after b^*

2) that doesn't contain substring aa.

$\{\lambda a, ab, ba, bb, aba, bba, bbb, bab, \ldots\}$

$R.E = ab^* + b^*a + b^*$ X

$= (a+b)b^*(a+b)$ X ababab.

$= b^*(ab^+)^* + a$ X ba.

$= b^*(ab^+)^* + a + (b^+a)^*$ X aba.

$= (ab)^* + (ba)^*(b)^*$ X babb

$= b^* + a(ba)^*b^* + b(ab)^* + a$ It think ok

R.E $\Rightarrow$ $(b + ab)^*(\lambda + a)$ ← but simplified is

Precedence of Operators:

$*$ L to R

$+$

$\cdot$

$+/|$

,1100,

)*

Date: ______ Day: M T W T F S

→ No of a's = No of b's.

$(ba)^*(ab)^* + a(ba)^*b.$

$(ab+ba+a(ba)^*b+b(ab)^*a)^*.$

(a^*b^+)

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

not possible ⇒ no regular language.

Proving Lang through recursive def

$\Sigma = \{a, b\}.$

Regular Lang = RL ⇒ $\{\Sigma, \lambda\}$

1) By default all Σ and λ are part of our language.

2) If r1 and r2 $\in$ RL then $r1^*$, $r1.r2$, $r1+r2$ belongs to RL.

$a^+ = a.a^*$

TOA-6 (30/01/24)

Automation model ⇒ to construct language.

Finite Automata.

Types

یقینی 1) Deterministic FA (DFA)

غیر یقینی 2) Non-deterministic FA (NDFA/NFA)

DFA = { Σ , Q , F , q_0 , δ }

- Σ → Alphabets
- Q → states (more than 1 possible)
- F → final states
- q_0 → initial state (only one)
- δ → transition function.

$\Sigma = \{0, 1\}$

$Q = \{q_0, q_1, q_2\}$

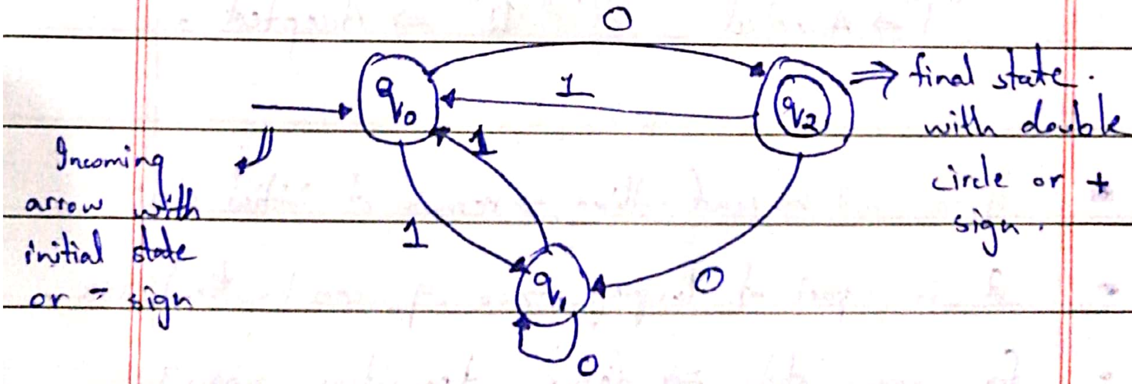
$F = \{q_2\}$.

q_0 = initial state.

Transitions. (Random)

States \ Σ	0	1
q_0	q_2	q_1
q_1	q_1	q_0
q_2	q_1	q_0

Transition Function:



q_0^- ⇒ initial state

q_2^+ ⇒ final state

If (input string after processing ends on final state):
accepted
else: rejected.

1) 0100 ⇒ ends on q_1 ⇒ rejected.
2) 110 ⇒ ends on q_2 ⇒ accepted.
q_0 $q_1 q_0 q_2$

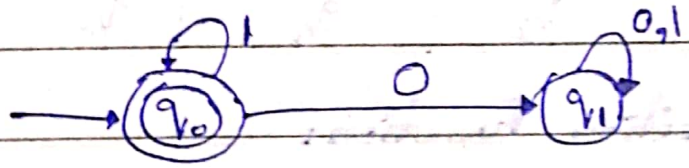
- DFA ⇒ Every letter has corresponding transition.
- NFA ⇒ Atleast one letter has no/multiple transitions.

Constructing Models:

$$\Sigma = \{0,1\}.$$

$$L = \{\lambda, 1, 11, 111, \ldots\}$$

$$R.E = 1^*$$



"1" ⇒ Accepted "11" ⇒ Accepted.

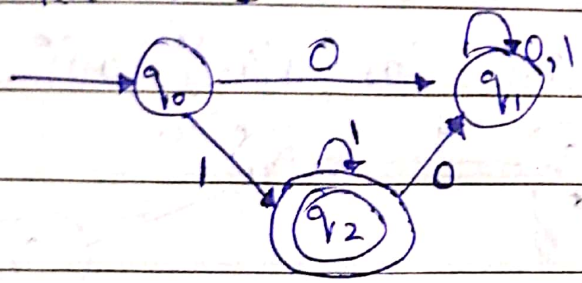
- When null ⇒ read nothing ⇒ remain at initial state.
- λ is part of language ⇒ so q_0 can be final state.
- For every state ⇒ define transition against each letter in sigma.

Date: Day: M T W T F S

- for 0 (in this case) move to state that never moves to final state as 0 is not acceptable.

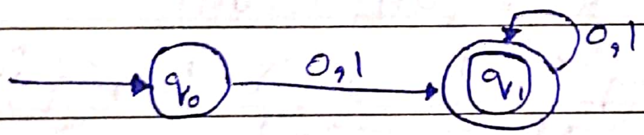
→ L2 = {1, 11, 111, ...}.

R.E = 1^+



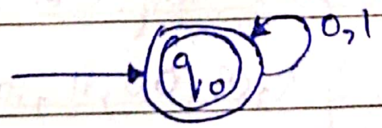
→ L3 = {0, 1, 01, 00, 10, ...}

R.E = $(0+1)^+$



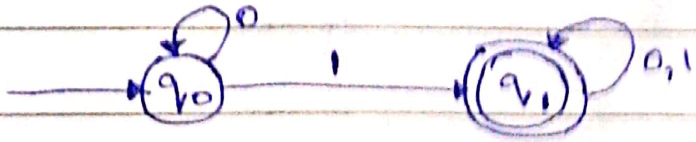
→ L4 = {λ, 0, 1, 01, 10, ...}

R.E = $(0+1)^*$



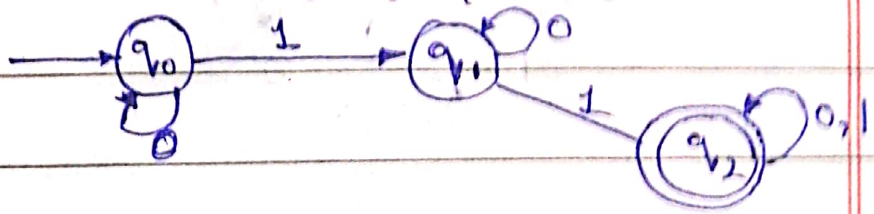
→ **L5** = Atleast one 1

R.E = $(0+1)^* 1 (0+1)^*$



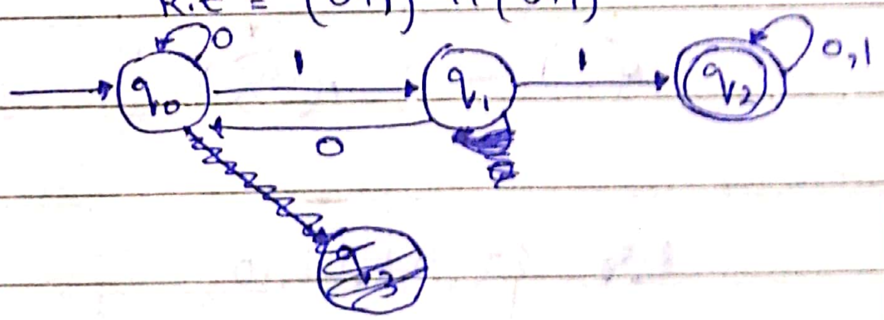
→ **L6** = Atleast two 1's

R.E = $(0+1)^* 1 (0+1)^* 1 (0+1)^*$



Extra **L** = consecutive two 1's.

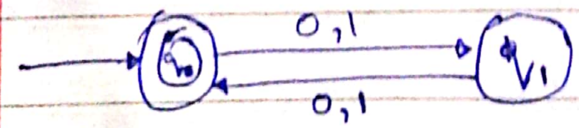
R.E = $(0+1)^* 11 (0+1)^*$



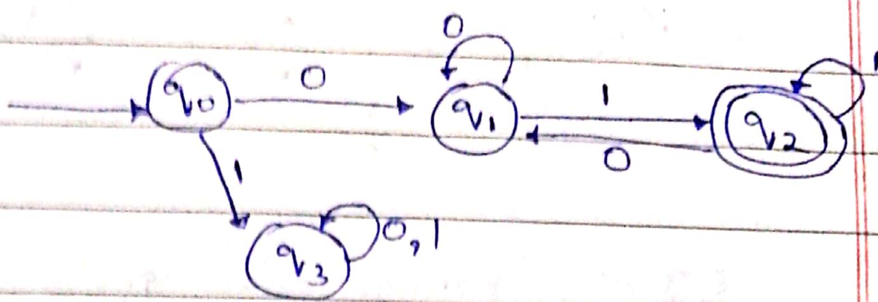
→ **L7** = Even length of string.

= { λ, 01, 00, 11, 10, ... }

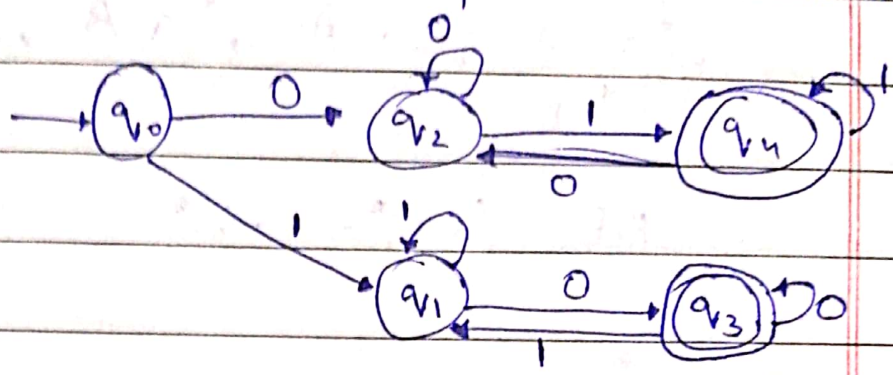
R.E = $((0+1)(0+1))^*$



→ **L8** = start with 0, end on 1.



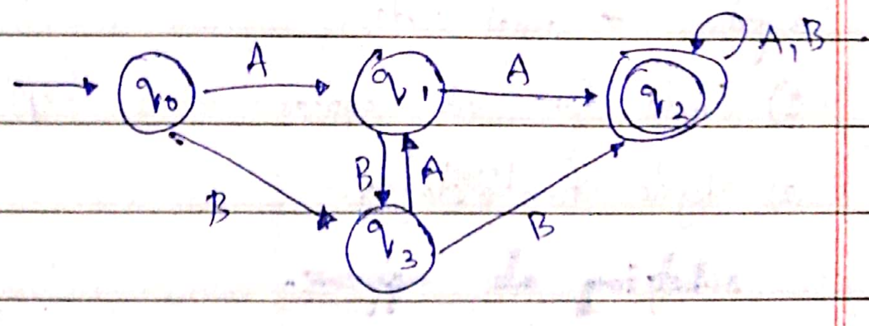
→ **L9** = start & end with different letter.



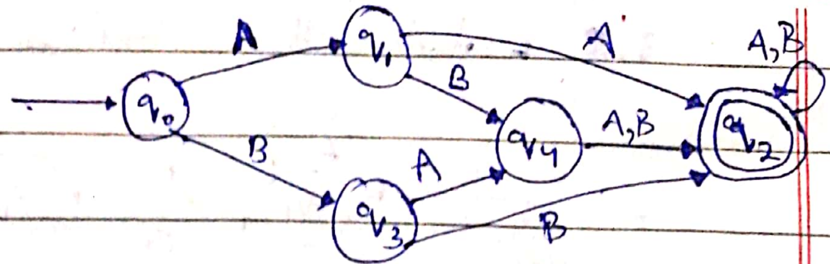
$\Sigma = \{a, b\}$ **TOA - 7** **(12/02/23)**

L1 = 2 consecutive A's or B's

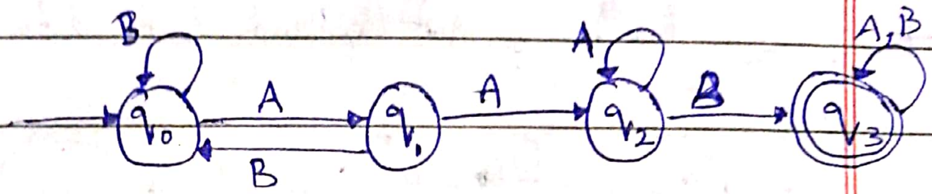
R.E = $(a+b)^*(aa+bb)(a+b)^*$



Extra → ~~Atleast~~ 2 A's or B's.



L2 = substring "aab" appears.



$R.E = (a+b)^* aab (a+b)^*$

L3 = All string without ab as substring.

$R.E = b^* a^*$.

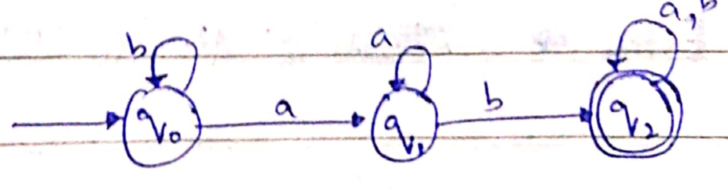
→ for negation DFA

first create for simple then negation of final states

1) Create for ab appear.

2) Negate final states.

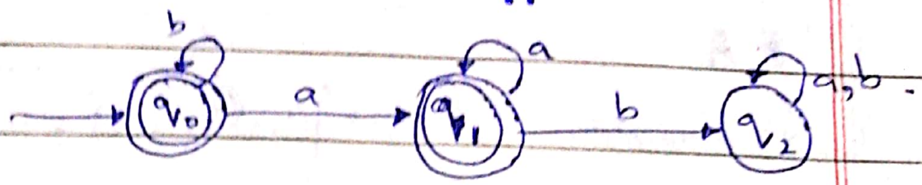
substring ab appear.



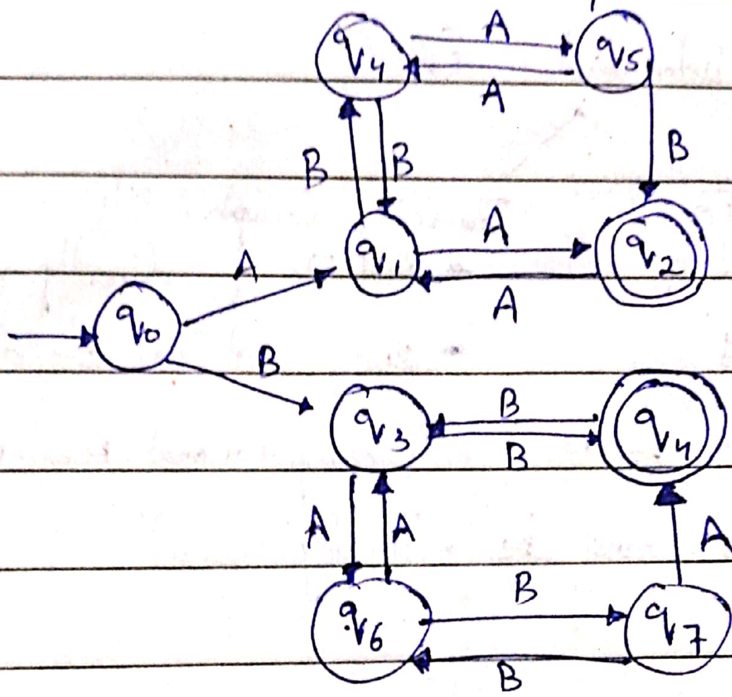
Date:

Day: M T W T F S

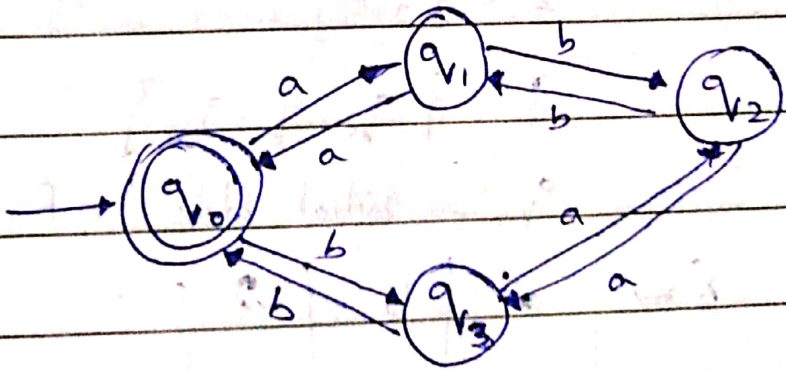
substring ab not appear



L4 ⇒ Even no of A's & Even no of B's



null not accepted.



null accepted

Date: Day: M T W T F S

TOA-8 (13/02/24)

F.A

1) Deterministic FA $\rightarrow \{\Sigma, Q, q_0, F, \delta\}$

↳ only 1 transition

↳ no state can be skipped.

2) Non-Deterministic FA

NFA / Transition Graph.

NFA:
- only 1 q_0 directly

Transition Graph:
- multiple q_0 directly defined.
- string input can be defined.

⇒ NFA → never on computational resources

⇒ NFA must be ultimately converted to DFA.

NFA

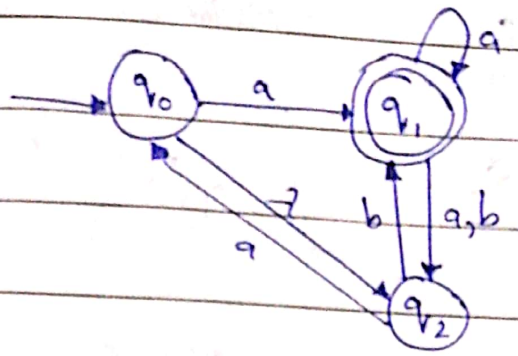
$\Sigma = \{a, b\}$

$Q = \{q_0, q_1, q_2\}$

$q_0 \Rightarrow$ Initial State $F = \{q_1\}$

$\delta \Rightarrow$	a	b	λ/ε
q_0	q_1		q_2
q_1	$q_1 q_2$	q_2	
q_2	q_1	q_1	

- missing transition + multiple transitions + can move without reading anything.



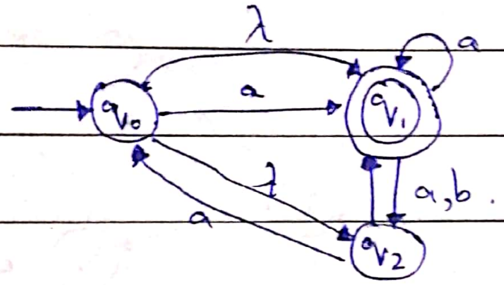
In NFA, same input can be processed by multiple paths

→ Is "b" acceptable? Yes first read λ.

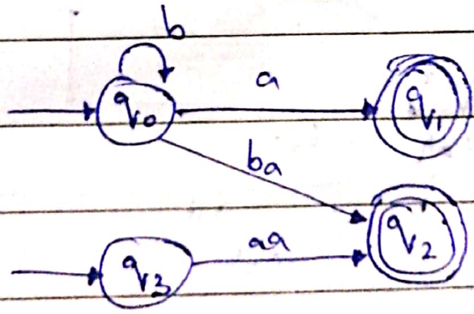
2 paths: $q_0 \Rightarrow$ path crashed.

$q_2 \Rightarrow$ accepted. ✓

↳ Atleast one path to be acceptable



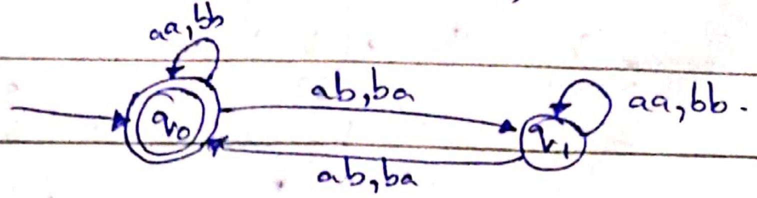
Transition Graph:



(directly more than 1 q_0 possible)

Date: ____________ Day: M T W T F S

L ⇒ Even no of A's & even no B's.

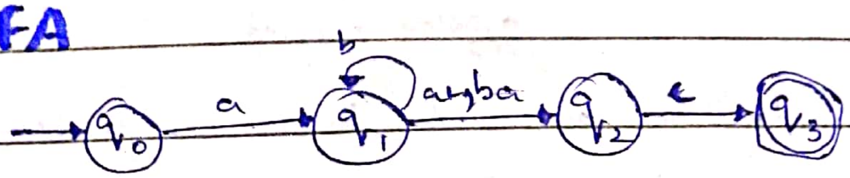


Construct DFA & NFA from R.E

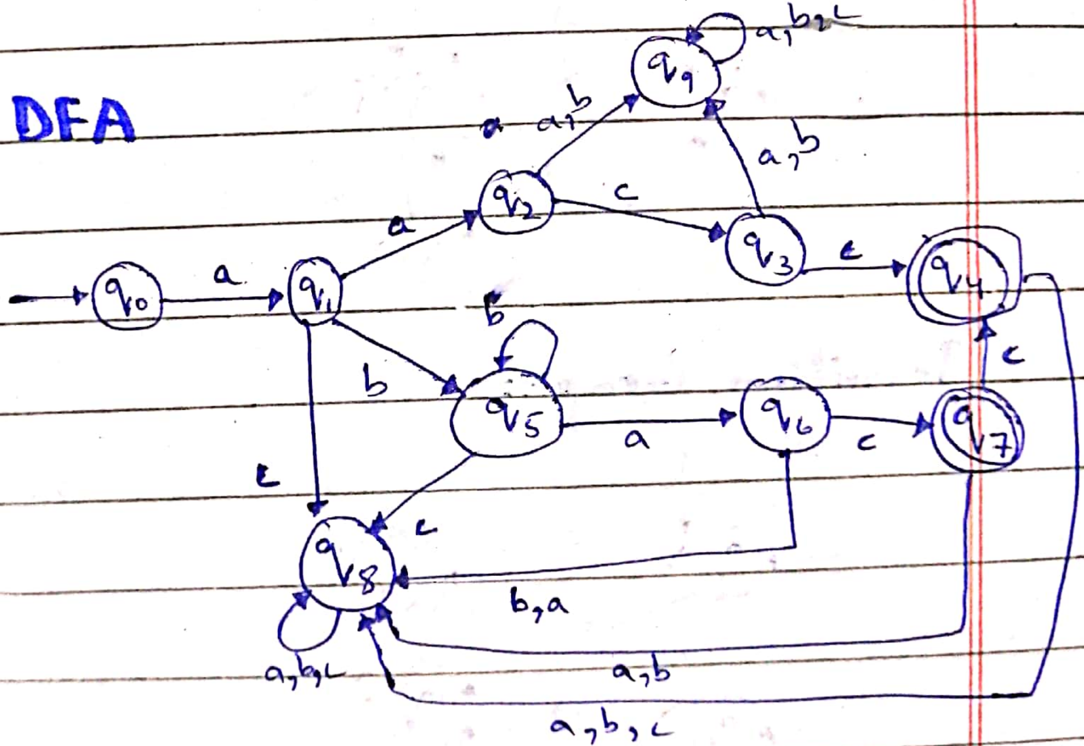
$$\Sigma = \{a, b, c\}$$

$$R.E = ab^*(ac+ba)c.$$

NFA



DFA



Date: Day: M T W T F S

Kleen's Theorem:

To convert NFA to DFA → activity to perform

Reg language ⇒ R.E ⇒ DFA ⇒ NFA ⇒ Transition Graph.

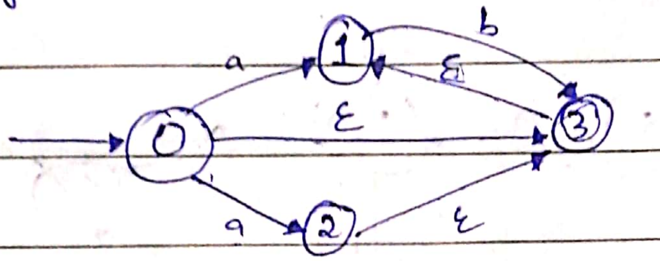
@ confirm 1) T.G can be converted to R.E

2) R.E can be used to generate NFA/DFA

3) NFA → T.G

NFA → DFA

- By subset construction method:



ε-closure:

{given all ε, λ transitions}

ε-closure $\{0\} \Rightarrow \{0,1,3\} \Rightarrow S_0$

$\{1\} \Rightarrow \{1\}$

$\{2\} \Rightarrow \{1,2,3\}$

$\{3\} \Rightarrow \{1,3\}$

Perform transition $(S_0, a) \Rightarrow \{1, 2\}$ {as {(1, 2), ...

1) ε-closure $(S_0, a) \Rightarrow \{1,2,3\} \Rightarrow S_1$

↳ from e.c {1}

& e.c {2}

2) ε-closure $(S0, b) \Rightarrow \{3\} \Rightarrow \{1,3\} \Rightarrow S2$.

3) ε-closure $(S1, a) \Rightarrow \{\} \Rightarrow S3$

4) ε-closure $(S1, b) \Rightarrow \{3\} \Rightarrow \overset{\{1,3\}}{S2}$. same as S2

5) " $(S2, a) \Rightarrow \{\} \Rightarrow S3$ same as S3

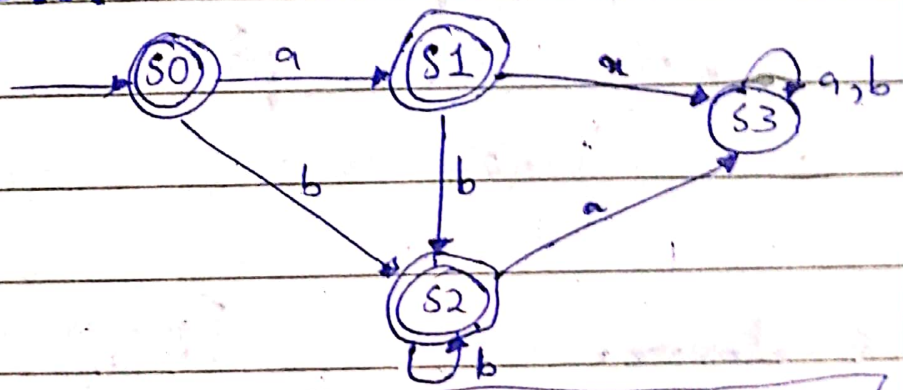
6) " $(S2, b) \Rightarrow \{3\} \Rightarrow \overset{\{1,3\}}{S2}$. same as S2

optional

7) " $(S3, a) \Rightarrow \{\} \Rightarrow S3$.

8) " $(S3, b) \Rightarrow \{\} \Rightarrow S3$

DFA



Since 3 is final state in NFA &

3 is in set S0, S1, S2 $\Rightarrow$ final state

TOA - 9 (19/02/24)

Regular language is represented by either

i) TG ii) FA iii) RE

↳ if represented by one can be converted to other form.

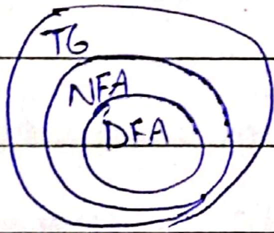
i) FA → TG

Kleen's Theorem } ii) TG → R.E

iii) R.E → FA

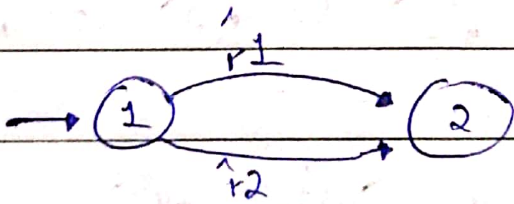
★ Prove these 3 parts

1) FA → TG

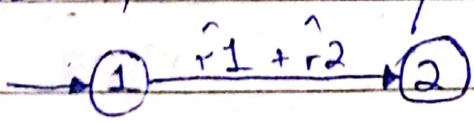


No efforts: As FA is also TG

2) TG → R.E

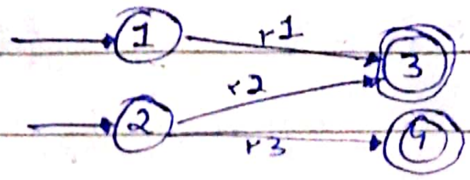


⇒ If multiple paths from one state to another, can be represented as:

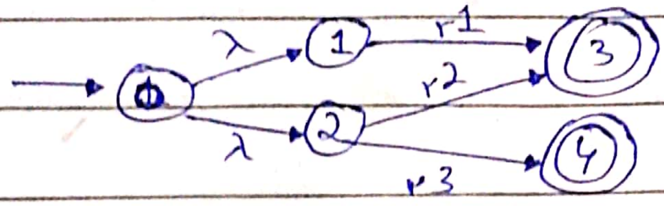


1) ⇒ Resolve multiple initial states.

(by adding 1 state before them).

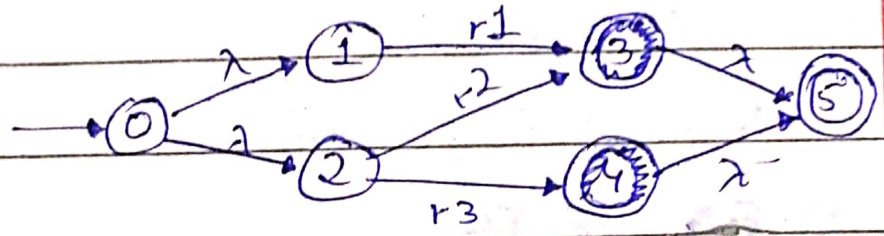


Sol:



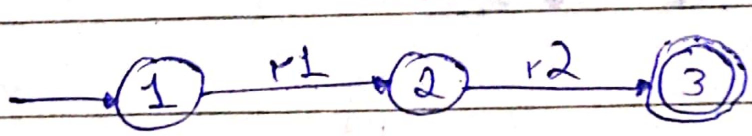
2) ⇒ Resolve multiple final states ⇒ only 1
(by adding 1 state at end)

Sol:



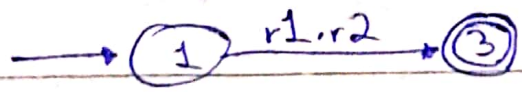
3) → ~~Resolve~~ Remove mid states 1-by-1.
↳ multiple scenarios possible.

1)

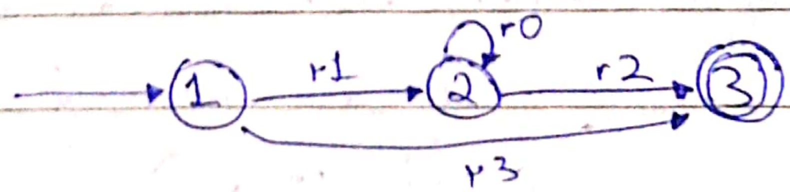


remove 2

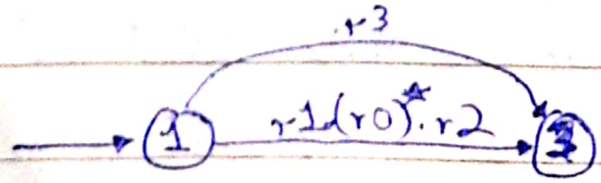
Sol:



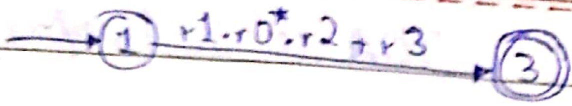
2)



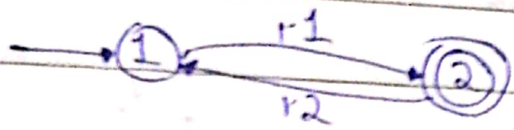
Sol:



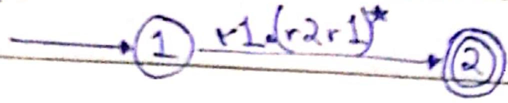
Date: ________ Day: M T W T F S



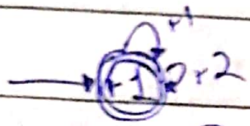
3)

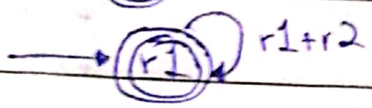


Sol:



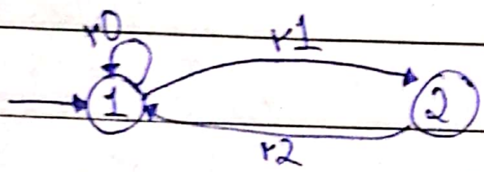
4)



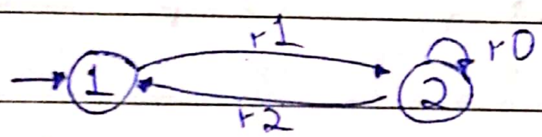


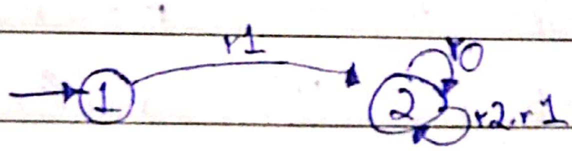
$R.E = (r1 + r2)^*$

5)



$R.E \Rightarrow r0^*.r1.(r2.r0^*.r1)^*$



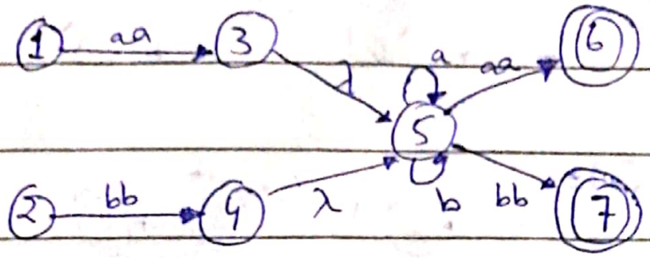


$R.E \Rightarrow r1.(r0+r2.r1)^*$

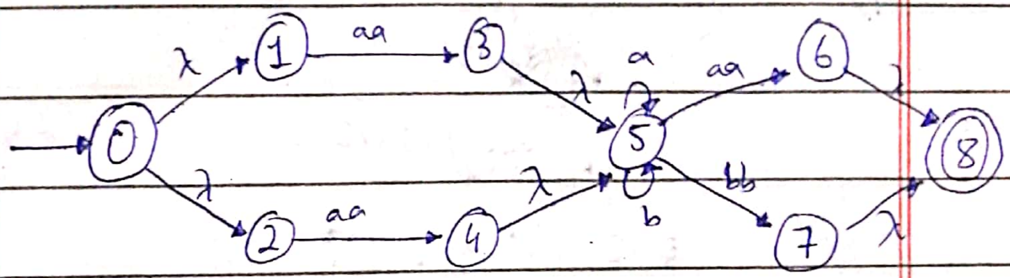
TOA-10 (20/02/24)

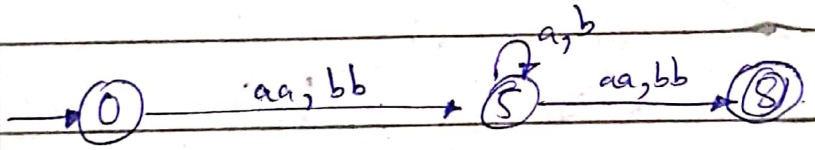
TG → R.E

1)



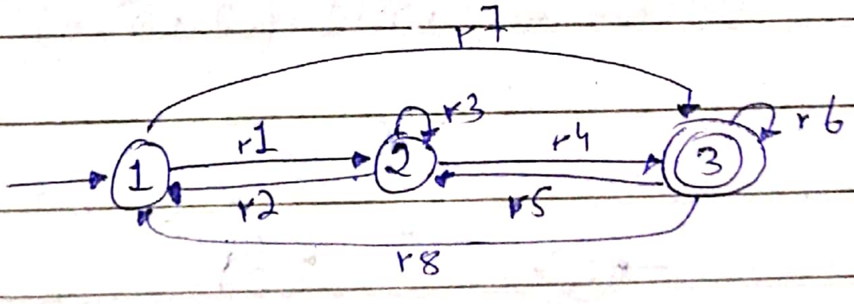
Sol:





R.E $(aa+bb).(a+b)^*.(aa+bb)$

2)

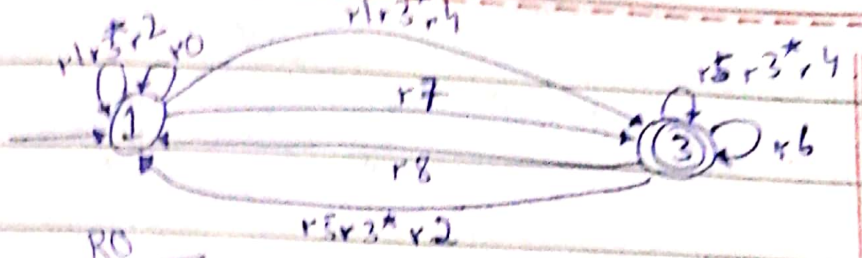


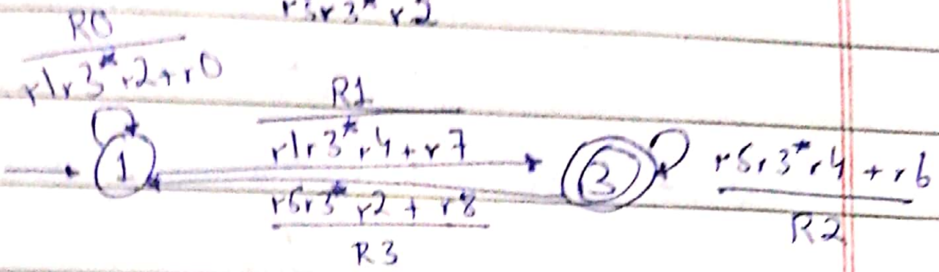
1-2-1 ⇒ 1-1 $r1\,r3^*\,r2$

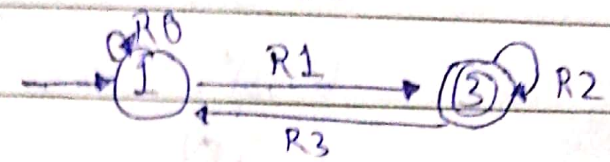
1-2-3 ⇒ 1-3 $r1\,r3^*\,r4$

3-2-3 ⇒ 3-3 $r5\,r3^*\,r4$

3-2-1 ⇒ 3-1 $r5\,r3^*\,r2$



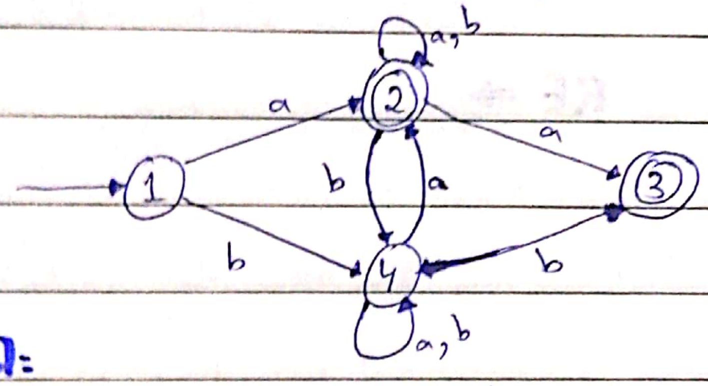




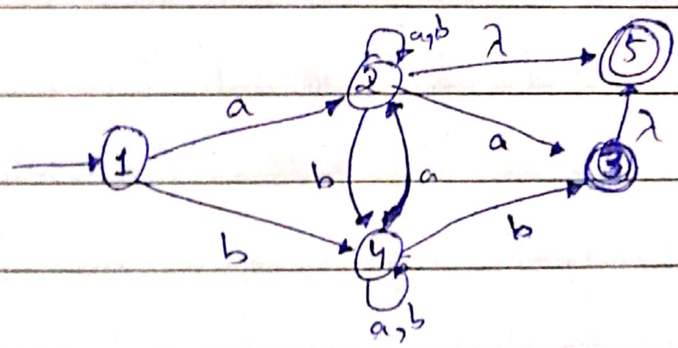
R.E ⇒ $R0^{*} R1 \left(R2 + R3\, R0^{*} R1\right)^{*}$

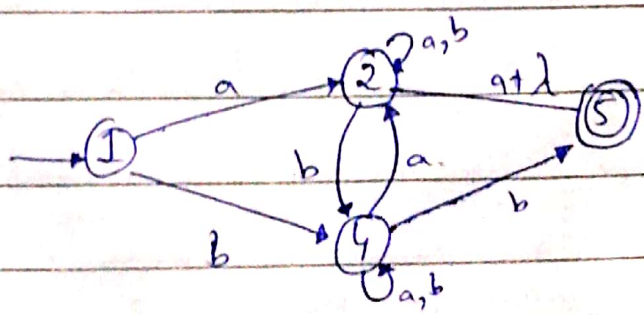
↳ put values

3)

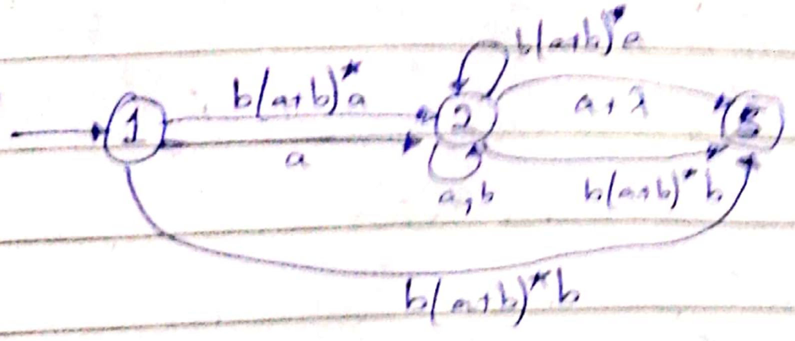


Sol:

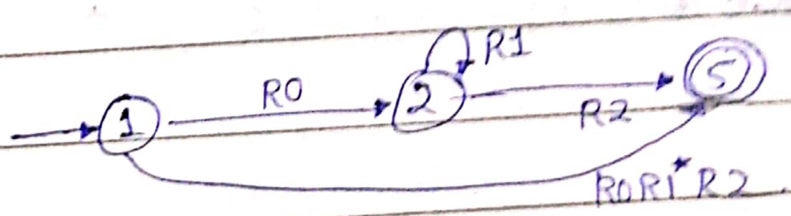




Date: Day: M T W T F S



1-4-5	—	1-5	b(a+b)*b
1-4-2	—	1-2	b(a+b)*a
2-4-2	—	2-2	b(a+b)*a.
2-4-5	—	2-5	b(a+b)*b.



RE ⇒

Date: ______ Day: M T W T F S

TOA-11 (26/02/24)

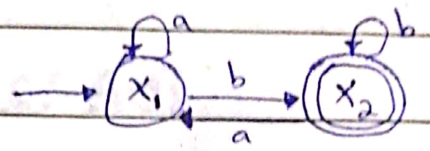
Kleen's Algo

3) RE → FA(DFA)

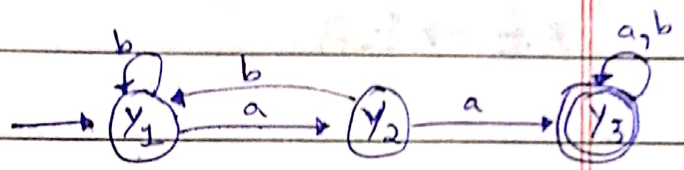
RL ⇒ 1) Union 2) Concatenate 3) Closure (* +)

string end with b	L1 :	R1	$(a+b)^*b$	DFA1
string contain aa	L2 :	R2	$(a+b)^*aa(a+b)^*$	DFA2

DFA1:



DFA2:



L1 ∪ L2

R1 ∪ R2

$(a+b)^*b + (a+b)^*aa(a+b)^* \Rightarrow L3$

⇒ Same steps for DFA.

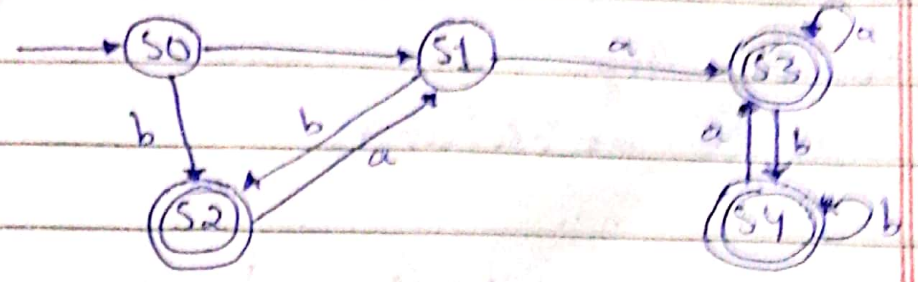
DFA1 ∪ DFA2.

	a	b
	x,a y,a	
(Initial states) → $S_0(x_1, y_1)$	$S1(x_1, y_2)$	$S2(x_2, y_1)$
$S1(x_1, y_2)$	$S3(x_1, y_3)$	$S2(x_2, y_1)$
Final $S2(x_2, y_1)$	$S1(x_1, y_2)$	$S2(x_2, y_1)$
Final $S3(x_1, y_3)$	$S3(x_1, y_3)$	$S4(x_2, y_3)$
Final $S4(x_2, y_3)$	$S3(x_1, y_3)$	$S4(x_2, y_3)$

$\boxed{x_2 \Rightarrow \text{final} \quad \text{or} \quad y_3 \Rightarrow \text{final}}$

Date: Day: M T W T F S

DFA:



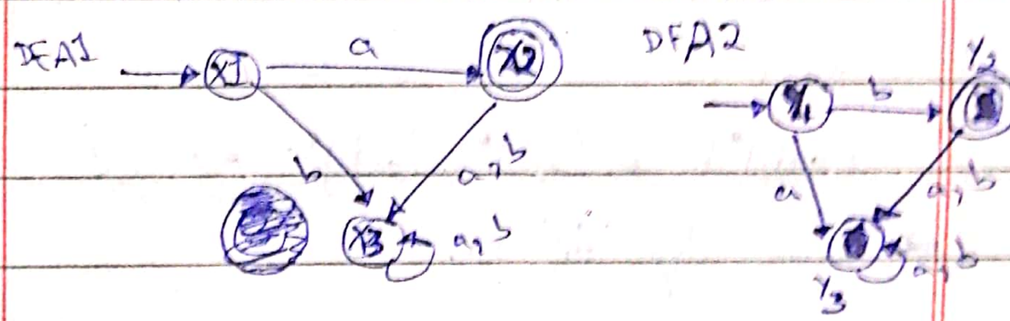
aab ? ✓ abb ? ✓

TOA-12 (27/02/24)

R.E → F.A

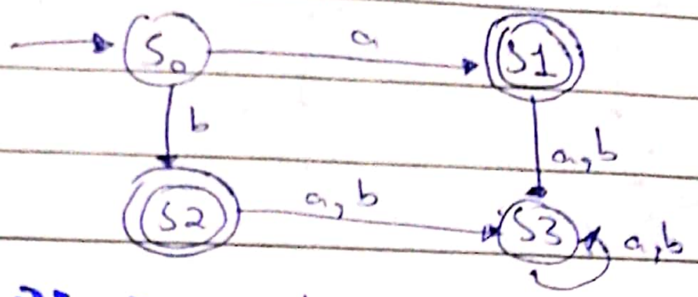
1) RE → a + b (union)

RE1: a, RE2: b



		a	b
	$S0(x_1, y_1)$	$S1(x_2, y_3)$	$S2(x_3, y_2)$
final	$S1(x_2, y_3)$	$S3(x_3, y_3)$	$S3(x_3, y_3)$
final	$S2(x_3, y_2)$	$S3(x_3, y_3)$	$S3(x_3, y_3)$
	$S3(x_3, y_3)$	$S3(x_3, y_3)$	$S3(x_3, y_3)$

DFA

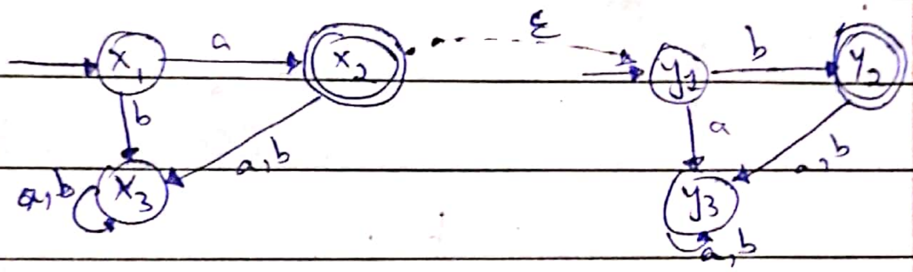


2) **RE ⇒** a . b (concatenate)

RE1 . RE2

initial state of left R.E & final state of Right RE

DFA1: DFA2:

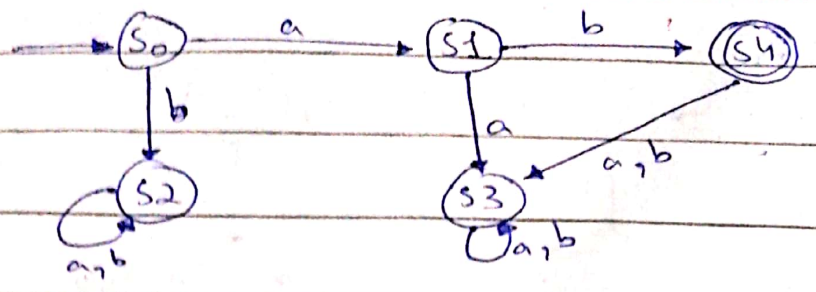


	a	b
$S_0(x_1)$	$S1(x_2, y_1)$	$S2(x_3)$
$S1(x_2, y_1)$	$S3(x_3, y_3)$	$S4(x_3, y_2)$
$S2(x_3)$	$S2(x_3)$	$S2(x_3)$
$S3(x_3, y_3)$	$S3(x_3, y_3)$	$S3(x_3, y_3)$
final. $S4(x_3, y_2)$	S3	S3

when reached 1st DFA's final state 2nd Auto include DFA's initial state

2nd DFA's final state ⇒ final state

Date: ____________ Day: M T W T F S

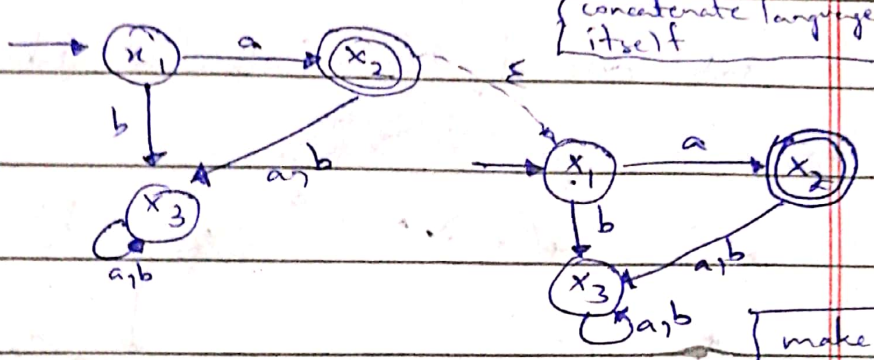


3) R.E $\Rightarrow (\underline{a})^*$ RE 1

3- closure $(*)$

$a^+ = a.a^*$

concatenate language with itself



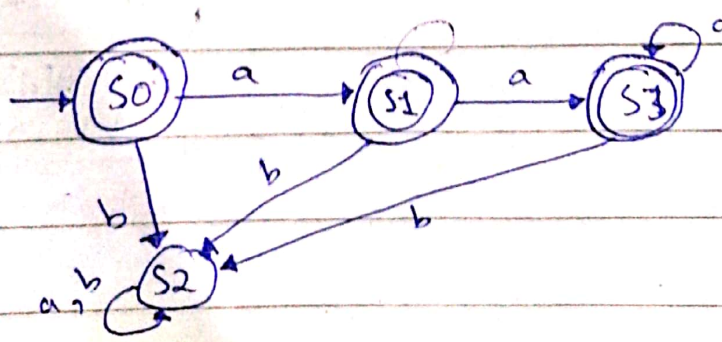
make inital state as final also

		a	b
F	$S_0(X_1)$	$S1(X_2, X_1)$	$S2(X_3)$
F	$S1(X_2, X_1)$	$S3(X_3, X_2, X_1)$	$S2(X_3)$
	$S2(X_3)$	$S2(X_3)$	$S2(X_3)$
F	$S3(X_3, X_2, X_1)$	$S3(X_3, X_2, X_1)$	$S2(X_3)$

$(x_3, x_3) \Rightarrow (x_3)$

$S1(X_2, X_1) \Rightarrow X_1, a \Rightarrow S1(X_2, X_1)$

means always write write x_1 with x_2



Exceptional Case: In closure, if initial state is again formed after transition, then don't repeat S_0 multiple time, rename new states.

R.E → DFA

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

L1: Odd length of string.

L2: contain atleast one a.

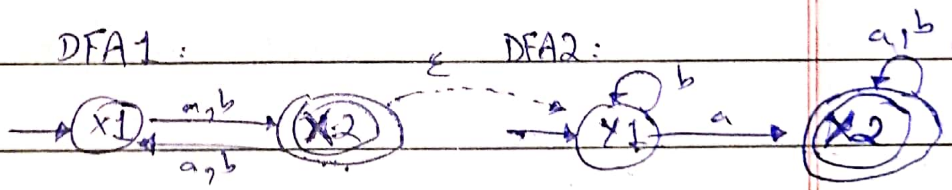
L1 . L2

R1. $(a+b).(aa+bb+ab+ba)^*$

R2. $(a+b)^* a (a+b)^*$

~~DFA~~: R1 . R2 =

DFA1 : DFA2 :

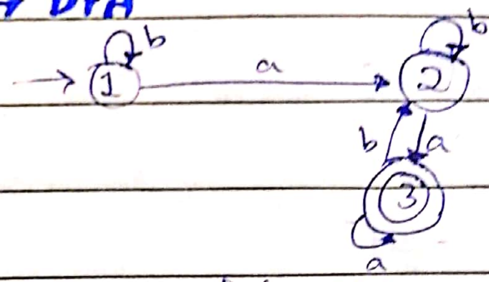


	DFA :	a	b
	$S_0(x_1)$	$S1(x_2, y_1)$	$S1(x_2, y_1)$
	$S1(x_2, y_1)$	$S2(x_1, y_2)$	$S3(x_1, y_1)$
Final	$S2(x_1, y_2)$	$S4(x_2, y_2)$	$S4(x_2, y_2)$
	$S3(x_1, y_1)$	$S4(x_2, y_2)$	$S1(x_2, y_1)$
Final	$S4(x_2, y_2)$	$S2(x_1, y_2)$	$S2(x_1, y_2)$

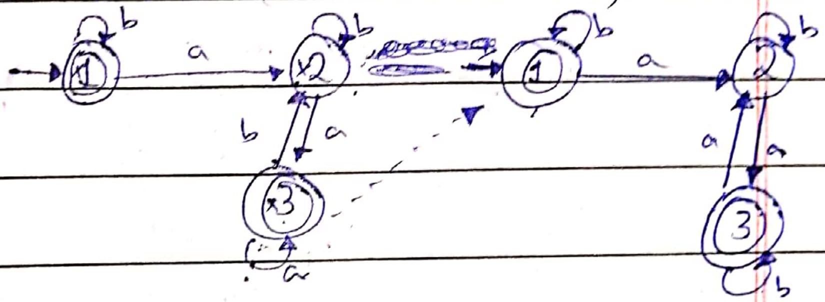
→ don't use same vocabulary again & again.

TOA-13 (04/02/24)

R.E → DFA



⇒ perform closure (concat with itself) of given DFA.

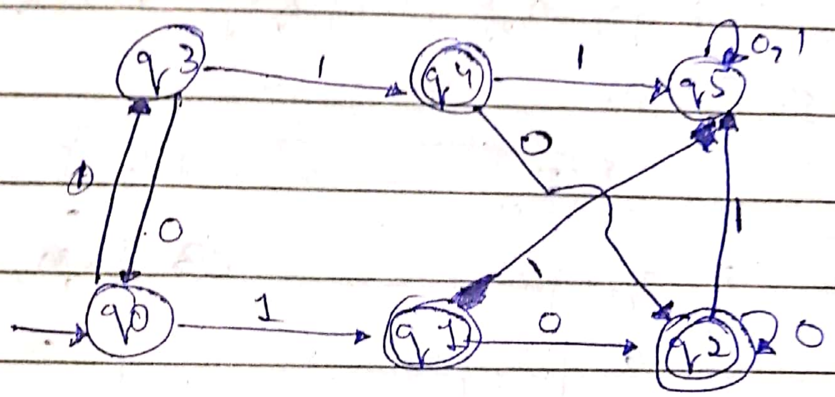


	DFA	a	b	
Final	$S_0(x1)$	$S_1(x2)$	$S_2(x1)$	repeated but with d/f name.
	$S_1(x2)$	$S_3(x3, x1)$	$S_1(x2)$	
	$S_2(x1)$	$S_1(x2)$	$S_2(x1)$	
final	$S_3(x3, x1)$	$S_4(x3, x1, x2)$	$S_5(2, 1)$	
final.	$S_4(x3, x1, x2)$	$S_4(x3, x1, x2)$	$S_5(1, 2)$	
	$S_5(2, 1)$	$S_4(2, 3, 1)$	$S_5(1, 2)$	

Date: Day: M T W T F S

⇒ Represent R.L ⇒ valid through model
⇒ convert the models (Kleen's theorem).
⇒ DFA optimization (reduce states).

DFA Optimization/Minimization.



1) Divide into 2 groups (final & non-final)

G1 G2

$\{q_0, q_3, q_5\}$ $\{q_1, q_2, q_4\}$

2)

0- Equivalence

	0	1
q_0	G1	G2
q_3	G1	G2
q_5	G1	G1

	0	1
q_1	G2	G1
q_2	G2	G1
q_4	G2	G1

which states are behaving same & which are behaving d/f?

⇒ all states behaving same

⇒ q_0 & q_3 are behaving same

1) No-splitting
2) Each state in separate group } ⇒ stop

Date: ______ Day: M T W T F S

∵ q5 is behaving d/f, so it will be further divided into 2 groups.

final

G1 {q_0, q_3} G2 {q_5} G3 {q_1, q_2, q_4}

1's Equivalence

$\boxed{\text{G2's don't need to be tested as only 1 state behav}}$

G1 {q_0, q_3}

	0	1
q_0	G1	G3
q_3	G1	G3

(no-split)

G2 {q_5}

	0	1
q_5	G2	G2

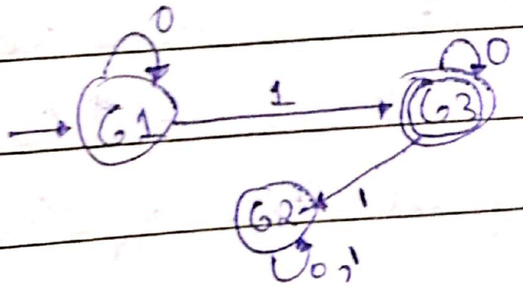
G3 {q_1, q_2, q_4}

	0	1
q_1	G3	G2
q_2	G3	G2
q_4	G3	G2

(non-split)

⇒ same groups ⇒ states converged into 3 state G1, G2, G3

⇒ $\boxed{\begin{array}{l}\text{Group with initial} \Rightarrow \text{initial} \\ \text{Group with final} \Rightarrow \text{final}\end{array}}$



Date: Day: M T W T F S

TOA - 14 (05/02/24)

Finite Automata with Output

i) Moore's Machine. $\{\Sigma, Q, q_0, F, \delta\}$

ii) Melay's Machine FA without output

⇒ FA with output machine ⇒ No final state.

$\{\Sigma, Q, q_0, \Gamma, \delta\}$

Tau

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

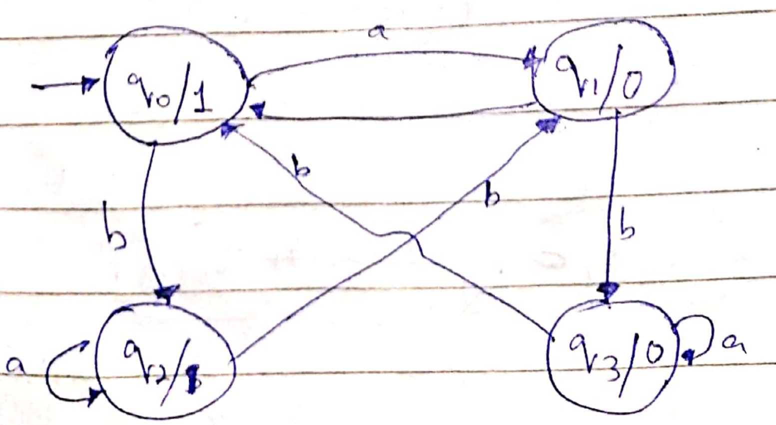
$Q = \{q_0, q_1, q_2, q_3\}$

$q_0 \Rightarrow$ int st

$\Gamma \Rightarrow \{0, 1\}$

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

i) Moore's Machine



input:	abbab
Output	100100

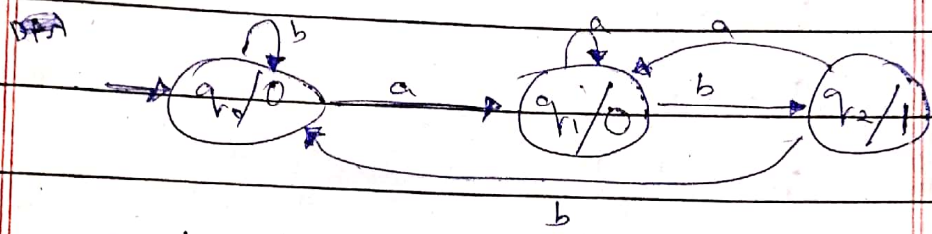
↳ At initial state

Q: machine should produce
⇒ certain output when pattern is meet

↳ let suppose when substring ab occur produce 1 as output.

⇒ baabb
00010

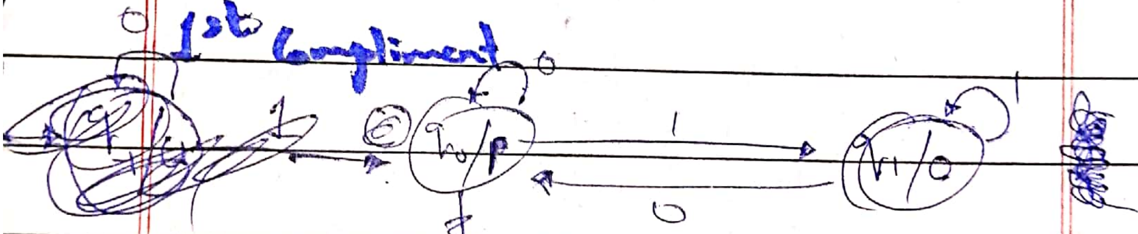
DFA



In	bba
Out	0000

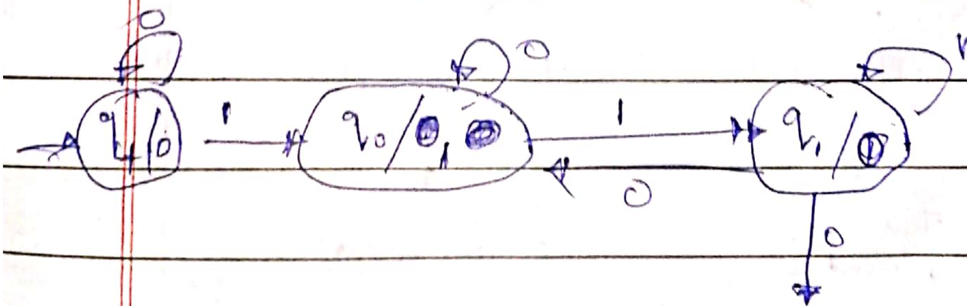
In	aabbab
out	0001001

1st Compliment



2's Compliment

~~1~~0100 ←
~~0~~1100
Invert bit after 1st 1.

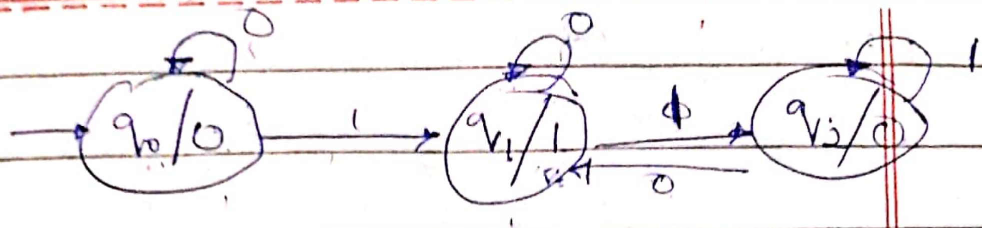


010100
→011011

001010
110110

010
011

Date. Day: M T W T F S

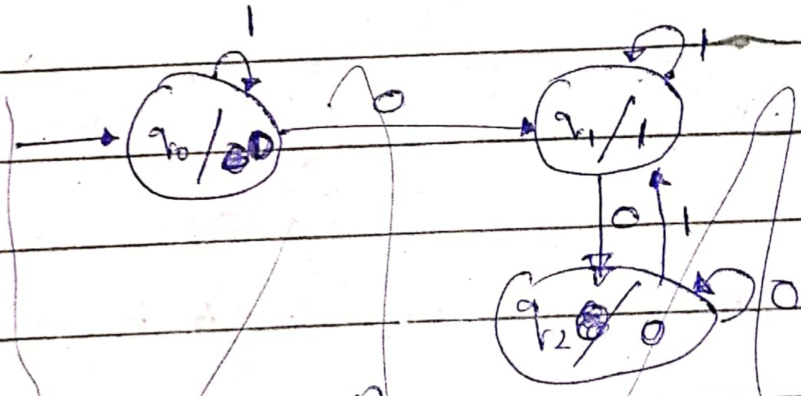


00101
a+ 0001010

00101
00011

3) Addition of 1 ⇒ same no of bits

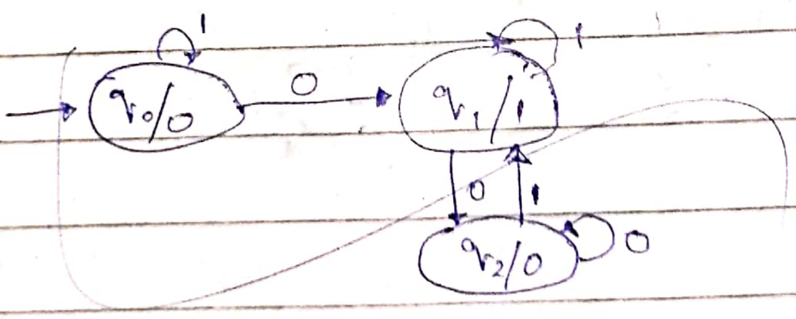
1011	100	111
+ 1	+ 1	+ 1
1100	101	(1) 000

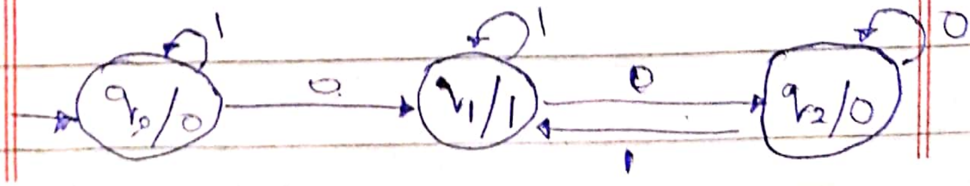


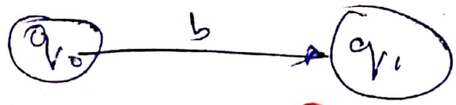
1011

110
11101
⇒ 0011

001
101
0101





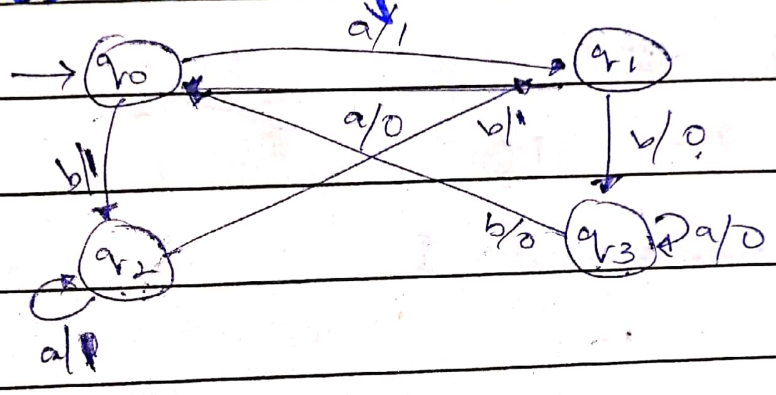


ii) Melay's Machine

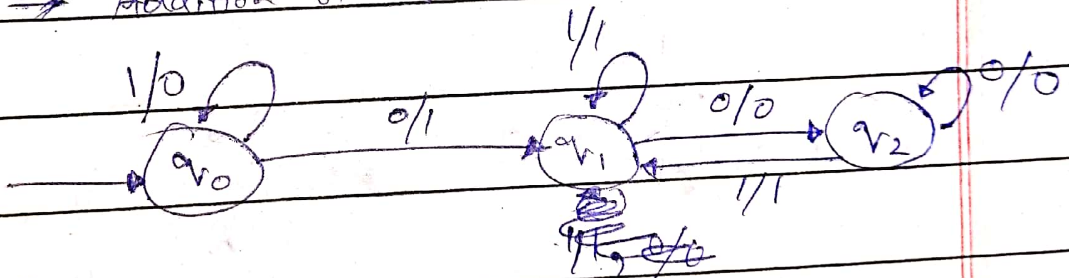
⇒ Moore machine produce extra output.

⇒ output on transition.

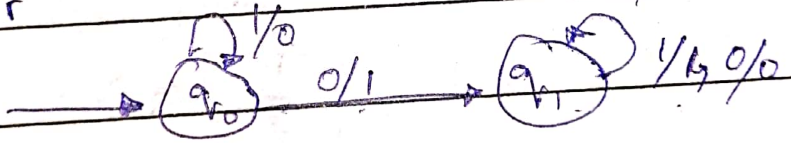
Moore's to Melay



⇒ Addition of 1.

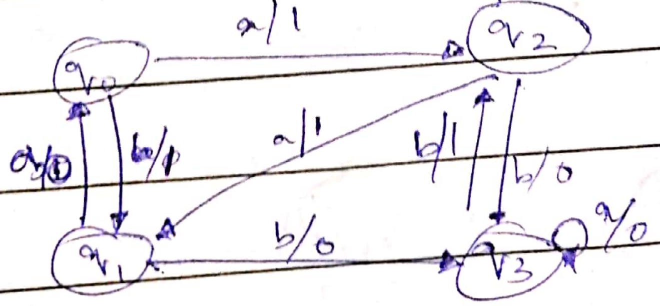


or



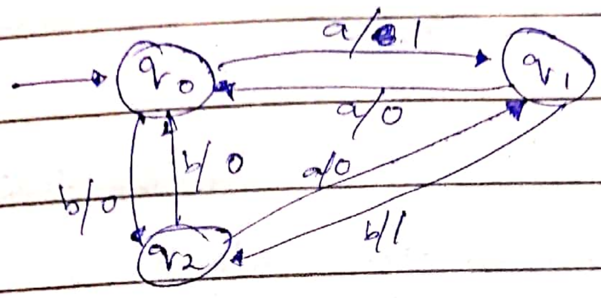
Moore → Mele

Moore to Melay



Date: Day: M T W T F S

Mealy to Moore



		a	b
$0 \Leftarrow$	q_0	$q_1, 1$	$q_2, 0$
$q_{10}, q_{11} \Leftarrow$	q_1	$q_0, 0$	$q_2, 1$
$q_{20}, q_{21} \Leftarrow$	q_2	$q_1, 0$	$q_0, 0$

$\Rightarrow$ split states on which output vary

		a	b
$0 \leftarrow$	q_0	q_{11}	q_{20}
$0 \leftarrow$	q_{10}	q_0	q_{21}
$1 \leftarrow$	q_{11}	q_0	q_{21}
$0 \leftarrow$	q_{20}	q_{10}	q_0
$1 \leftarrow$	q_{21}	q_{10}	q_0