Unit ⇒ 1

Q.1) (a) Difference between NFA and DFA are :-

NFA	DFA
→ Full form: Non determinsitic finite automata.	Full form: Determincitic finite automata.
→ Dead configuration is allowed.	Dead configuration is not allowed.
→ ε-move is allowed.	ε-move is not allowed
→ All real computers are not non deterministic.	→ Real computers are deterministic.
→ Designing and understanding is easy.	→ Designing and understanding is not easy compare to NFA.
→ No. of state is less.	→ No. of state is more.

Hence, this are the several Difference between Non-deterministic finite automata and Deterministic finite Automata.

1 (b) To construct Moore Machine equivalent to Melay Machine :-

	a=0		a= 1	
	State	Output	State	Outputs
→ q_1	q_1	1	q_2	0
q_2	q_4	1	q_4	1
q_3	q_2	0	q_3	1
q_4	q_3	0	q_2	1

⇑ Moore Machine Transitions:-

∵ We have state output as:-

$q_1 = 1$

$q_2 = 0$

$q_3 = 0$

$q_4 = 1$

Moreover on Moore Machine,

Transition $(\delta) \rightarrow$: ~~Q×~~

$$\delta : Q \rightarrow \Delta$$

where, Δ = Output symbol

Q = Total set of states

& In Melay Machine

$$\delta : Q \times \Sigma \rightarrow \Delta$$

where, :-

Σ = Input symbols
ρ = Set of States
Δ = output symbol

so when we'll convert Moorey Machine equivalent to Melay Machine by the help of its Transitions then;

States	a=0 States followed by output	a=1 States followed by output
q_1	q_1, 1	q_2, 0
q_2	q_4, 1	q_4, 1
q_3	q_2, 0	q_3, 1
q_4	q_3, 0	q_2, 1

⟸ Melay Machine

Hence this is the required equivalent to Melay Machine from Moore Machine through its Transition rules.

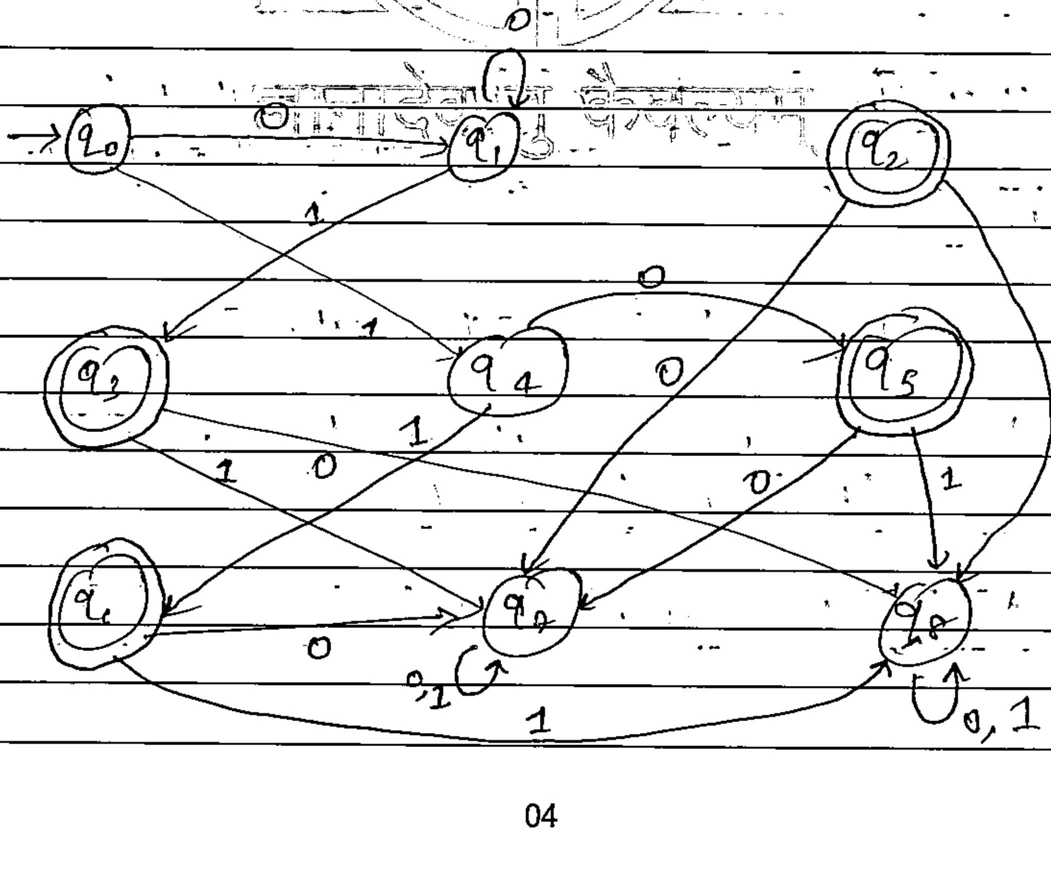
∴ We got to from here is that:

→ Melay Machine's Output depends on the present state as well as input states.

→ Asynchronous in nature.

1C) Minimize the DFA :-

State/Σ	0	1	
→q0	q1	q4	
q1	q1	q3	
q2	q7	q8	×
q3	q8	q7	
q4	q5	q6	
q5	q7	q8	
q6	q7	q8	
q7	q7	q7	
q8	q8	q8	



① from q_0 we can go → $q_1, q_4, q_5, q_6, q_3, q_7, q_8$,
But we can't go to q_2 --- So it get elementtated

from q_1 we can go → q_3, q_8 & else we can't go anywhere so we'll use step-2 :-

Now devide the states into 2 classes :-

Accepting Classes	Non accepting Classes
$\lambda_0 = \{q_3, q_5, q_6\}$ A_1	$\{q_0, q_1, q_4, q_7, q_8\}$ A_2

Now take q_0 & q_1 & compare it in a table & look whether they comes in a same group or not :-

$q_0 \to q_1, q_4$ hence their output doesnot comes
$q_1 \to q_2, q_3$ in same group.
& for q_0 if we'll check it with other states also we'll conclude that their output doesnot belong to same group so we'll seprate it.

⇒ $\{q_0\}$

Likewise same goes for q_1 it also get's seperated as $\{q_1\}$

Then, q_4 also get seperated on other class.

So,

$$\{q_4\}$$

for q_7 & q_8 We conclude their output comes in a same group so they have their combined class.

$$\{q_7, q_8\}$$

Now,

Non accepting class =

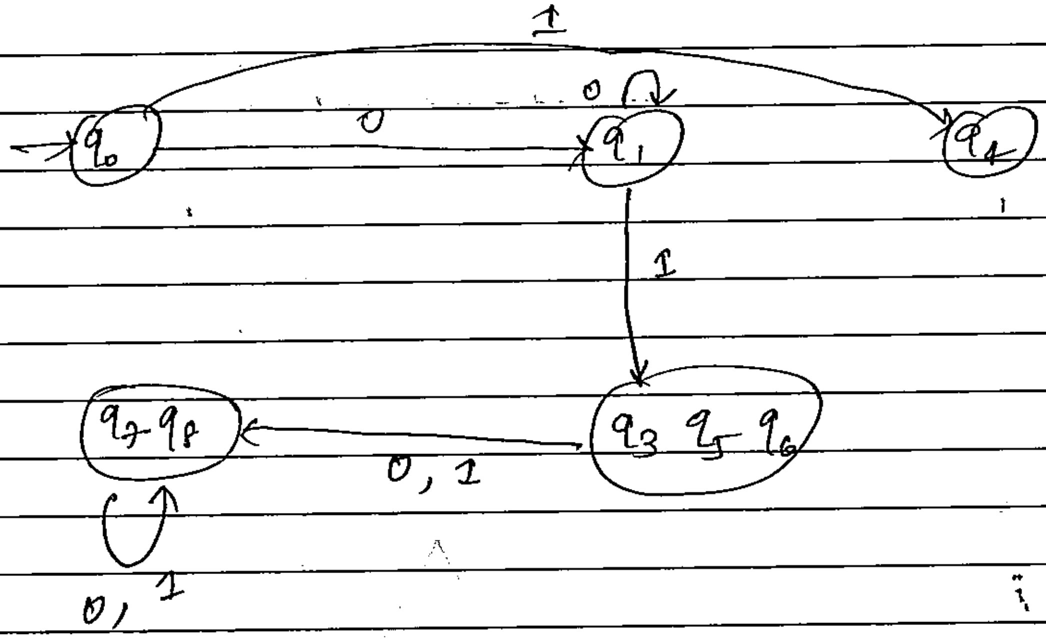
$$\{q_0\}\ \{q_1\}\ \{q_4\}\ \{q_7, q_8\}$$

for q_3 & q_5 & q_6 we conclude they come in same group i.e of non acception class.

$$\{q_3, q_5, q_6\}$$

Hence,

$$\lambda_0 = \underbrace{\{q_0\}}_{Q_{11}}\ \underbrace{\{q_1\}}_{Q_{12}}\ \underbrace{\{q_4\}}_{Q_{13}}\ \underbrace{\{q_7, q_8\}}_{Q_{14}}.\ \underbrace{\{q_3, q_5, q_6\}}_{Q_{15}}$$



Hence this is Minimized DFA.

Unit-2

Q2) (a) Closure property of regular grammer:-

Here when we'll get finite string then we'll use (+).

eg. $\Sigma(a,b)$, $\{aa, ab, ba, bb)$

So, Can be written as $(a+b)(a+b)$

Kleen closure is * we used for n numbers of possibilities.

→ Complements Can be used.

→ Substitution & replacement of the symbos also we can do.

as by substituting or replacing one symbol Of any language with other language.

2C) A/Q,

To prove the following language is not regular:-

For this we'll use pumming lemma's concept.

Hence, Pumping lemma States that:-

Consider L as a language from which take w (String) that should be greater than n. which further divides in to 3 parts x, y & z.

for $i \geq 0$, xy^iz

$|y| > n$

$|xz| < n$

Now By considering or following this we'll solve the given language.

∵ We have;

$$L = \{a^n b^n \text{ where } n \geq 1\}$$

⇒ let the given language is a ~~past~~ Regular language.

∴

$$L = a^n b^n$$

now, Take any String from it as:-

let the value of $n = 3$ so;

$$L = a^3 b^3$$

i.e aaabbb

Hence this is the String (w) = aaabbb which also follows the given condition so,

Now divide w into 3 parts x, y, z :-

$$\underbrace{a}_{x}\ \underbrace{a\ a}_{Y}\ \underbrace{b\ b\ b}_{z}$$

We have :- $x = a$

$Y = aa$

$z = bbb$

P.T.O

Now, we'll pump y :-

Let i be equal to ~~φ~~ 2 So;

We have;

$x = a$

$y = aa$

$z = bbb$

Then $y = aaaa$ $\left(\because xy^2z\right)$

Now,

our language becomes:-

$$\boxed{aaaaabbb}$$

as our language should contain equal no's of a and b but Here both are unequal.

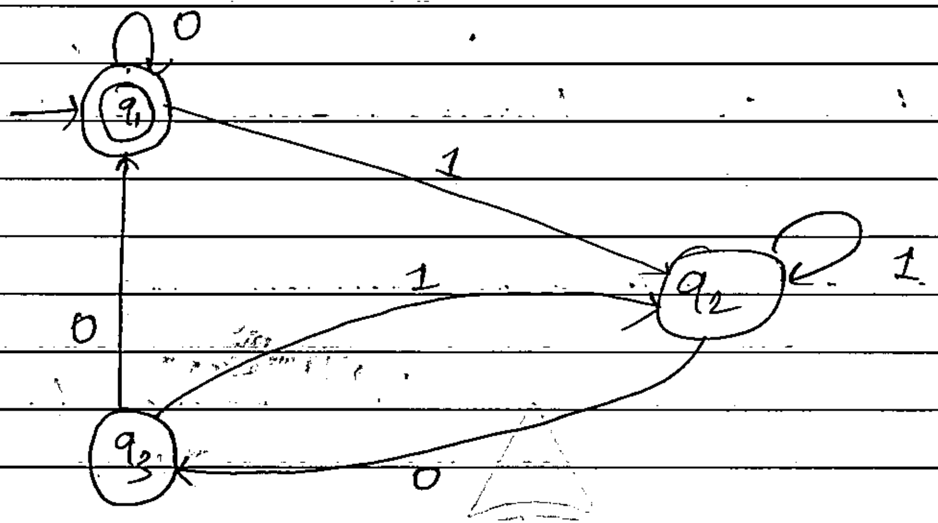
So we conclude that :-

$$\boxed{aaaaabbb \notin L}$$

So, the given language is Not Regular.

Hence proved

2. (d) A/Q,



To convert transition system to Regular expression We'll use Arden's theorem:-

$$\boxed{\begin{array}{l} R = Q + RP \\ R = QP^* \end{array}} \longrightarrow \text{Arden's statement}$$

Now we'll make the equation for each states:-

$$q_1 \longrightarrow q_1 0 + q_3 0 + \varepsilon \quad - \text{①}$$

$$q_2 \longrightarrow q_1 1 + q_2 1 + q_3 1 \quad - \text{②}$$

$$q_3 \longrightarrow q_2 0 \quad - \text{③}$$

P.T.O

Now put q_3 in eq (2) from eq (1)

$$q_2 = q_1 1 + q_2 1 + q_2 0 1 \quad \text{--- (4)} \quad (\because q_3 = q_2 0)$$

put eq^n (3) in (1) also we get;

$$q_1 = q_1 0 + q_2 0 0 + \varepsilon \quad \text{--- (5)} \quad (\because q_3 = q_2 0)$$

~~Now pr~~

We can write eq^n (4) as;

$$q_2 = q_1 1 + q_2 1 + q_2 0 1$$

$$q_2 = q_1 1 + q_2 (1 + 0 1)$$

$$\boxed{q_2 = q_1 1 (1+01)^*} \quad \text{--- (4)} \quad \left(\begin{array}{l} R = Q + RP \\ R = QP^* \end{array} \right)$$

put this eq^n in eq^n (5) we get;

$$q_1 = q_1 0 + q_1 1 (1+01)^* 0 0 + \varepsilon$$

$$q_1 = \varepsilon + q_1 \left(0 + 1 (1+01)^* 0 0 \right).$$

∵ It is also in the form of $R = Q + RP$

So, change it to $\boxed{R = QP^*}$:-

$$q_1 = \varepsilon + q_1\left(0 + 1(1+01)^*00\right)$$

⇓ could be written as

$$q_1 = \varepsilon\, q_1\left[\cancel{\varepsilon}0 + 1(1+01)^*00\right]^*$$

(when we put $\varepsilon = 1$)

So,

$\cancel{q_1 = q_1[0+1(10}$

Ans ⇒ $\boxed{q_1 = q_1\left[0 + 1(1+01)^*00\right]^*}$

Hence this is the regular Expression which we got from the following transition System:-

Where * sign tells you that n no. of possible is there for given expression.

P.T.O

Unit-3

Q3) a) (1) A/Q,
regular expression for the set of all strings having odd no. of 1's :-

~~(00)*1(01)*"~~

~~Here any No. of 0 can~~

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

(2)

~~00~~ $(0+1)^*00$ → Regular Expression

Hence this is the required string.

3b Chomsky Classification of Grammar are:-

① Type 0 Grammer.
② Type 1 Grammer
③ Type 2 Grammer
④ Type 3 Grammer.

⊛ Type 0 Grammer :-

→ also known as unrestricted gramma.

Here in this grammar :-

→ We should have atleast one variable in x.

$$\underbrace{x}_{\text{atleast one variable}} \longrightarrow \underbrace{Y}_{(V+T)^*}$$

→ Any no. of variable and Termination can come on Y.

eg. $\boxed{aA \longrightarrow BCab}$

⊛ Type 1 Grammer :-

→ also known as Regular Grammar.

Here in this grammar :-

→ Same as (T-0) grammer but length of x should be less than Y. (atleast one variable in x).

i.e. $\boxed{|x| \leq Y}$

→ Start Symbol can also contain Epsilon if required.

eg.

$$\boxed{aA \rightarrow abCD}$$

④ Type-2 Grammar:-

→ also known as Content free Grammer

→ comes in content free language

Here,

→ Only single variable should be there in x.

→ length of x should be less than the length of Y.

$$\boxed{|x| \le Y}$$

→ Y can contain any no. of variables & Terminator.

→ No restriction in condition of Epsilon.

$$S \rightarrow AB$$
eg. $$A \rightarrow bCD$$
$$B \rightarrow \varepsilon$$

(4) Type - 3 Grammer:-

→ This type of Grammer also contains single variable in x.

→ length of x should be less than the length of φ.

$$\boxed{|x| \leq |\varphi|}$$

→ φ can contain any no. of Terminators with a variable in its left most side or on its Right most side only.

→ If Variable will be in left most side then we called it as left most gramar.

→ If Variable will be on Right most side then we called it as Right most grammar.

eg. $$\boxed{\begin{array}{c} A \rightarrow aaaB \\ \text{or} \\ A \rightarrow Baaa \end{array}}$$

Hence, this is all about Chomsky classification of Grammar.

3d) Alg,

$$S \rightarrow 0B \mid 1A$$
$$A \rightarrow 0 \mid 0S \mid 1AA$$
$$B \rightarrow 1 \mid 1S \mid 0BB$$

put S in A we get;

$$S \rightarrow 0B \mid 1A$$
$$A \rightarrow 0 \mid 00B \mid 01A \mid 1AA$$
$$B \rightarrow 1 \mid 1S \mid 0BB$$

Now put S in B we get;

$$S \rightarrow 0B \mid 1A$$
$$A \rightarrow 0 \mid 00B \mid 01A \mid 1\overbrace{AA}^{\alpha}$$
$$B \rightarrow 1 \mid 10B \mid 11A \mid 0\underbrace{BB}_{B}$$

Then,

$$S \rightarrow 0B \mid 1A$$
$$A \rightarrow 0 \mid 00B \mid 01A \mid 1AA' \mid 1A$$
$$A' \rightarrow AAA' \mid AA$$
$$B \rightarrow 1 \mid 10B \mid 11A \mid 0BB' \mid 0B$$
$$B' \rightarrow BBB' \mid BB$$

Now put the value of each in the required productions:-

$S \rightarrow 01 \mid 010B \mid 011A \mid 0BB' \mid 00B \mid 10 \mid 00B \mid$
$101A \mid 11AA' \mid 11A$

$A \rightarrow 0 \mid 001 \mid 0010B \mid 0011A \mid 000BB' \mid 000B \mid$
$01A \mid 1AA' \mid 1A$

$A' \rightarrow$ ~~AA~~ $00A' \mid 00B00BA' \mid 01A01AA' \mid 1AA'$
$1AA'A' \mid 1A1AA' \mid 00 \mid 00BB \mid 01A01A \mid$
$1AA'1AA' \mid 1A1A$

$B \rightarrow 1 \mid 10B \mid 110 \mid 1100B \mid 1101A \mid 111AA' \mid$
$111A \mid 0BB' \mid 0B$

$B' \rightarrow BBB' \mid BB$

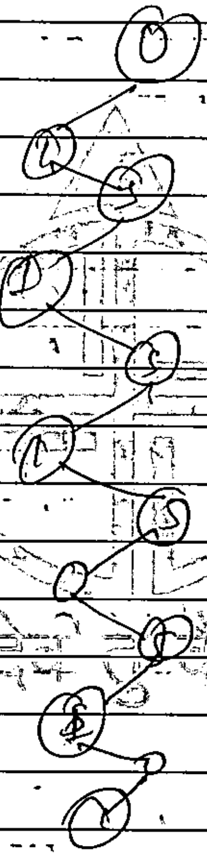
Now put the value of A' & B' we get;

$S \rightarrow 01 \mid 010B \mid 011A \mid 0B1 \mid 0B10B \mid 0B110 \mid$
$0B1100B \mid 0B1101A \mid 0B111AA' \mid 0B111A \mid$
$0B0BB' \mid 0B0B$

$A \rightarrow 0 \mid 001 \mid 0010B \mid 0011A \mid 000BB' \mid$
$000BB \mid 000B \mid 01A \mid 1A00A' \mid$

Thereafter we'll substitute the value of A' & B' to it.

Calculate the derivation tree for the string W = 00110101



Unit-4

Q4) (a) Diffrence between NPDA & DPDA are :-

NPDA	DPDA
→ Full-form:- Non-deterministic push down automata	Full form:- Deterministic push down automata.
→ ε-move is allowed	→ ε-move is not allowed.
→ Designing is easy	→ Designing is quite Hard compare to NPDA
→ understanding is easy	→ understanding is not so easy.
→ Number of states are less.	Number of states are more.
→ Dead configuration is allowed	Dead configuration is not allowed.

Hence, this are the several differences between NPDA & DPDA.

Q4) (b) A/Q,

Design a pushdown automata (PDA) which accepts the language:

$L = \{a^n b^{2n}\}$ where $n \geq 1$

By using input tapes and Stack We will design it

∵ We know that;

PDA consists of 7 tuples which are:-

$\{Q, \Sigma, \Gamma, \delta, Z_0, q_0, F\}$

∵ Q = Set of states

Σ = input Symbols

Γ = inputs in Stack (input that comes in Stack).

δ = Transition

Z_0 = Initial value already present on Stack.

q_0 = initial state

F = final state

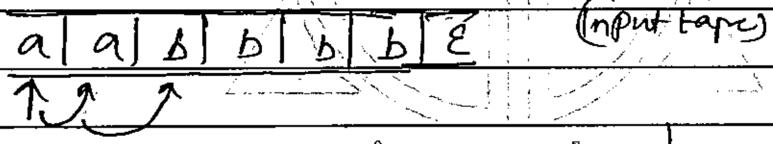
Now Alg;

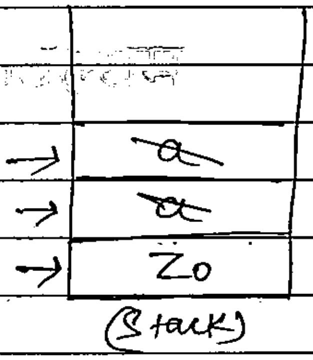
We have:- $L = \{a^n b^{2n}\}$ where $n \geq 1$

a	a	b	b	b	b	ε

→ logically if we'll think than on input tape we get this.

∵ b is 2 times of a.

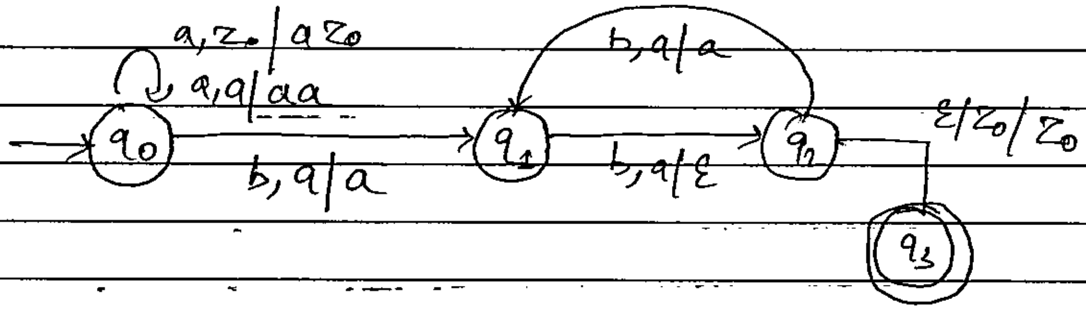




$q_0 a Z_0 \rightarrow q_0 a Z_0$
$q_0 a a \rightarrow q_0 a a$
$q_0 b a \rightarrow q_1 a$
$q_1 b a \rightarrow q_2 \varepsilon$
$q_2 b a \rightarrow q_1 a$ → Transitions
$q_2 b a \rightarrow q_2 \varepsilon$

$q_2 \varepsilon Z_0 \rightarrow q_3 Z_0$

⇒ P.T.O for State Diagram:-



Hence,

PDA: $\{(q_0, q_1, q_2, q_3), (a, b)\ \delta, (a, Z_0), \overset{Z_0,}{} q_0, q_3\}$

Ans

Hence this is the required PDA which accepts the language

4d (2) Halting problem of Turing Machine:-

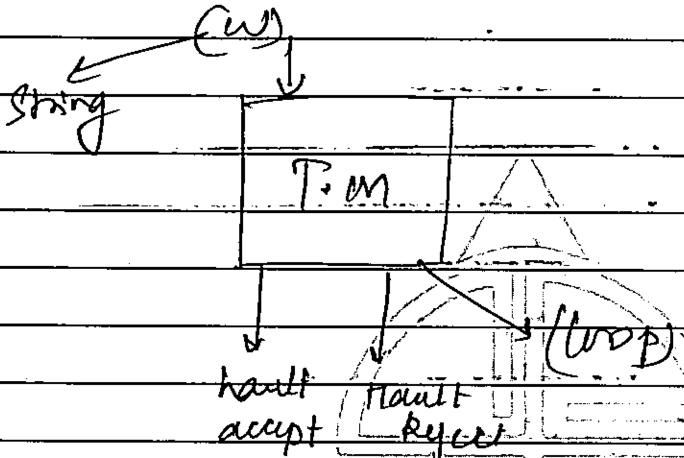
Halting problems occur in Recursive Enumerable language only.

∵ w.k.t. Turing Machine is designed to recognized a Recursive Enumerable language

In Short it is designed for it only.

In the Recursive Enumerable language we have 3 states :-

→ Hault and accept
→ Hault and Reject
→ No hault



→ In case of No hault, Turing Machine doesnot able to decide whether a language is Recursive Enumerable then at that time string under goes on a loop i.e upto infinite called as infinite Loop & the Haulting doesnot occurs.

→ In such cases :-
→ language don't gets accepted and not even rejected. means doesnot understand by the Turing Machine.

Moreover it is closed under the operation (— , & Complement.

4(a)) (1) Decidable and undecidable problem:-

2

(a) Decidable problem is nothing ~~big~~ but partially a Recursive numerable language.

→ which gets recognized by Turing Machine

→ Contains 3 states:-

① Hault and accept
② Hault and Reject
③ ~~Has~~ No Hault.

→ closed in operations '∪' & Complement.
→ It is solvable.

(b) undecidable problem:-

→ It is nothing but recursive language.

→ which gets recognized by full or Total turing machine.

→ contains 2 States:-

① Hault and accept
② Hault and Reject

→ closed in operations "Substitution" & "Replaced".

unit-05

Q5) (a) partial function:-

⇒ A function $F : x \rightarrow Y$ is defined for some values of x is known as partial functions.

eg. $F : x \rightarrow N$

$$\boxed{f(n) = n/2} \longrightarrow \text{case of partial function}$$

(b) initial function:

(i) Zero function:-

where $Z(n) = 0$ eg. $Z(5) = 0$
$\forall n \in N$

(ii) Successive function:-

where

$$S(x) = x + 1 \quad \forall n \in N$$

eg.

$$S(6) = 6 + 1 = 7$$

(iii) projection function:-

$$U_i^n (n_1, n_2, n_3 \dots n^n)$$

eg.

$$U_2^3 (1, 2, 3) = \textcircled{2}$$

Hence, this is all about initial function. Ans

5b) Ag:-

$f(x,y) = x*y$ & $f(x,y) = x^y$ is primitive recursive function

So,

For :-

$f(x,y) = x*y$.

Firstly take ~~Ø~~ $(y=0)$ so;

$$f(x,0) = x*y$$
$$= x*0$$
$$= 0$$
$$= Z(x)$$

Then, put $(y=y+1)$ so;

~~$x($~~ ~~$f(x, y+1) = x*(y+1)$~~

$$f(x,y+1) = x*y + x$$
$$f(x,y+1) = f(x,y) + x$$

now,

$$h(x, y, f(x,y)) = U_3^3(x,y,f(x,y)) + U_1^3(x,y,f(x,y))$$

$$= \boxed{h(x,y,z) = u_1^3(x,y,z) + u_3^3(x,y,z)}$$

Hence, it contains initial function so (*) or $f(x,y) = x*y$ is primitive recursive function.

Now for $f(x,y) = x^y$

Take $y=0$ so;

$$f(x,0) = x^0$$
$$= 1$$
$$= g(x)$$

Then put $y = y+1$ so;

$$f(x, y+1) = x^{y+1}$$
$$f(x, y+1) = x^y \cdot x$$
$$f(x, y+1) = f(x,y) \cdot x$$

$$h(x, y, f(x,y)) = u_3^3(x, y, f(x,y)) \cdot u_1^3(x, y, f(x,y))$$

$$\boxed{h(x,y,z) = u_1^3(x,y,z) \cdot u_3^3(x,y,z)}$$

∵ w.k.t, * is also primitive recursive so x^y is hence also a primitive recursive.

Hence Both are Primitive recursive function.

5 (d) **Computation:-**

→ If any system starts running with the specific function that mean it or system gets computed by the function This process is said to be computation

Also,

Turing Model for Computation:-

→ If functional for Natural Number gets computed by an algo if and only if the Turing machine also gets computed.

→ The work i.e been done by the Real computers could also be done By the Turing Machine.

→ Mathematic Module suggestion also given to the Turing machine but It doesn't requires any one of them because It has so much power.

→ There is No problem which can be solved by Real computer ~~but~~ an can-not by Turing Machine

∵ Turing Machine is so much powerful that it can solve the finite automata problems as well as push down automata problems.

Here, we can compare the three alphabets on a Specific language.

for eg.

$L = \{a^n b^n c^n\}$ where $n \geq 0$

→ Here we can Read as well as write.

→ we can move in any direction while solving the problem & uses Blank also.

Hence, this is the Turing model for computation. & last but not least it uses two stacks.

ज्ञानादेव तु कैवल्यम्