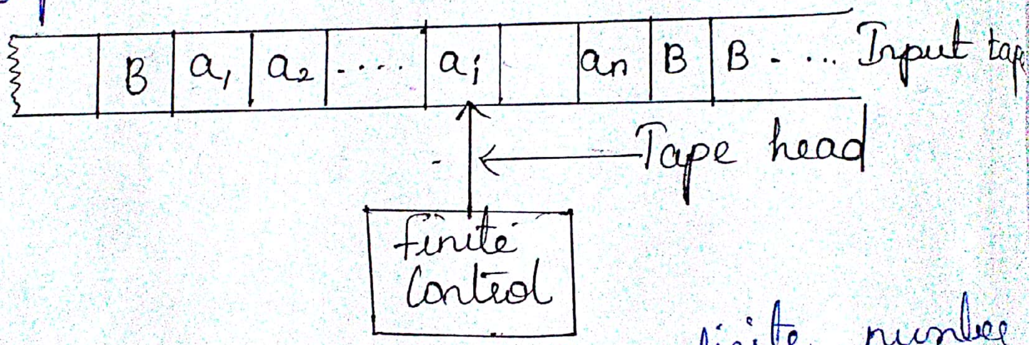
Introduction to Turing Machines

Turing machines:-

Alan Turing introduced a new mathematical model called Turing machine during the year 1936. It is mostly used to define languages and to computer integer functions.

The basic model has a finite control, an input tape that is divided into cells, and a tape head that scans one cell of the tape at a time



* Each cell can hold one of a finite number of tape symbols.
* All other tape symbols extending infinitely to the left and right hold a special symbol called blank.
* Initially the tape head is pointing the leftmost cell that holds the input.

In one move, the turing machine depending upon the symbol scanned by the tape head and the state of finite control

1) Changes its state
2) it prints a symbol on the tape cell scanned, replacing what was written there
3) Moves its head, one cell to left or right.

Definition :-
A turing machine M is a 7-tuple
$$M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$$
where
Q - finite set of states
Σ - finite set of input symbols
Γ — finite set of tape symbols
δ - the transition function.
The arguments of $\delta(q, X)$ are a state q and a tape symbol X. The value of $\delta(q, X)$ if it is defined, is a triple (p, Y, D), where
1. p is the next state in Q.
2. Y is the symbol in Γ, written in cell being scanned, replacing whatever symbol was there
3. D is a direction, either L or R, and telling us the direction in which the head moves

q_0 - The start state, a member of Q, in which the finite control is found initially
B - The blank symbol. This symbol is in Γ but not in Σ.
F - The set of final or accepting states, a subset of Q.

$\delta(q, a) = (p, o)$

Instantaneous Description for Turing Machine

The ID of turing machine is defined in terms of the entire input string and the current state.

We shall use the string $X_1 X_2 X_{i-1}\, q\, X_i\, X_{i+1} \ldots X_n$ to represent the ID in which,

1) q is the state of turing machine
2) the tape head is scanning the ith symbol from the left.
3. $X_1 X_2 \ldots X_n$ is the portion of the tape between leftmost and rightmost nonblank.

We describe the moves in turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ by the $\vdash_M$ notation that was used for PDA's. As usual, $\overset{*}{\vdash}_M$, or just $\overset{*}{\vdash}$ will be used to indicate, zero, one, or more moves of the TM M.

Suppose $\delta(q, X_i) = (P, Y, L)$ (ie) the next move is leftward. Then

$X_1 X_2 \ldots X_{i-1}\, q\, X_i\, X_{i+1} \ldots X_n \vdash_M X_1 X_2 \ldots X_{i-2}\, p\, X_{i-1}\, Y X_{i+1} \ldots X_n$

This move reflects the change to state p and the fact that the tape head is now positioned at cell $i-1$.

There are two important exceptions :-

1) If $i=1$, then M moves to the blank to the left of X_1. In that case,

$$X_1 X_2 \cdots X_{i-1}\, q\, X_i X_{i+1} \cdots X_n \vdash_M pBYX_2 \cdots X_n.$$

2) If $i=n$ and $Y=B$, then the symbol B written over X_n joins the infinite sequence of trailing blanks and does not appear in the next ID. Thus,

$$X_1 X_2 \cdots X_{n-1}\, q\, X_n \vdash_M X_1 X_2 \cdots X_{n-2}\, p\, X_{n-1}$$

Now suppose $\delta(q, X_i) = (p, Y, R)$; Then (ie) the next move is rightward.

$$X_1 X_2 \cdots X_{i-1}\, q\, X_i X_{i+1} \cdots X_n \vdash_M X_1 X_2 \cdots X_{i-1} Y p X_{i+1} \cdots X_n.$$

Here, the move reflects the facts that the head moved to cell $i+1$. Again there are two important exceptions :-

1) if $i=n$, then the $i+1$st cell holds a blank, and that cell was not part of the previous ID. Thus, we instead have

$$X_1 X_2 \cdots X_{n-1}\, q\, X_n \vdash_M X_1 X_2 \cdots X_{n-1} Y p B$$

2) If $i=1$ and $Y=B$, then the symbol B written over X_1 joins the infinite sequence of leading blanks and does not appear in the next ID. Thus,

$$q X_1 X_2 \cdots X_n \vdash_M p X_2 \cdots X_{n-1}$$

Pblm:-

Design a Turing machine that will accept a language $L = \{0^n 1^n / n \geq 1\}$

Sln:-

The TM will change a 0 to an X and then a 1 to a Y, until all 0's and 1's have been matched.

$\therefore M = (\{q_0, q_1, q_2, q_3, q_4\}, \{0,1\}, \{0,1,X,Y,B\}, \delta, q_0, B, \{q_4\})$

State q_0 is the initial state, and M also enters state q_0 every time it returns to the leftmost remaining 0.

If M is in state q_0 and scanning a 0, then M goes to state q_1 and change 0 to X and moves right.

Once in state q_1, M keep moving right over all 0's and Y's that it finds on the tape, remaining in state q_1.

If M sees an X or a B, it dies.

However if M sees a 1 when in state q_1 it changes that 1 to a Y, enters into q_2 and starts moving left.

In state q_2, when M reaches the rightmost X, which marks the right end of the block of 0's that have already been changed to X, M returns to state q_0 and moves right.

A Turing machine to accept $\{0^n 1^n \mid n \geq 1\}$

State	0	1	X (Symbol)	Y	B
q_0	(q_1, X, R)	–	–	(q_3, Y, R)	–
q_1	$(q_1, 0, R)$	(q_2, Y, L)	–	(q_1, Y, R)	–
q_2	$(q_2, 0, L)$	–	(q_0, X, R)	(q_2, Y, L)	–
q_3	–	–	–	(q_3, Y, R)	(q_4, B, R)
q_4	–	–	–	–	–

Example :- Input string 0011

Initial M ID is $q_0 0011$

Sequence

$q_0 0011 \vdash X q_1 011 \vdash X0 q_1 11$

$\vdash X q_2 0Y1 \vdash q_2 X0Y1$

$\vdash X q_0 0Y1 \vdash XX q_1 Y1$

$\vdash XXY q_1 1 \vdash XX q_2 YY$

$\vdash X q_2 XYY \vdash XX q_0 YY$

$\vdash XXY q_3 Y \vdash XXYY q_3 B$

$\vdash XXYYB q_4 B$

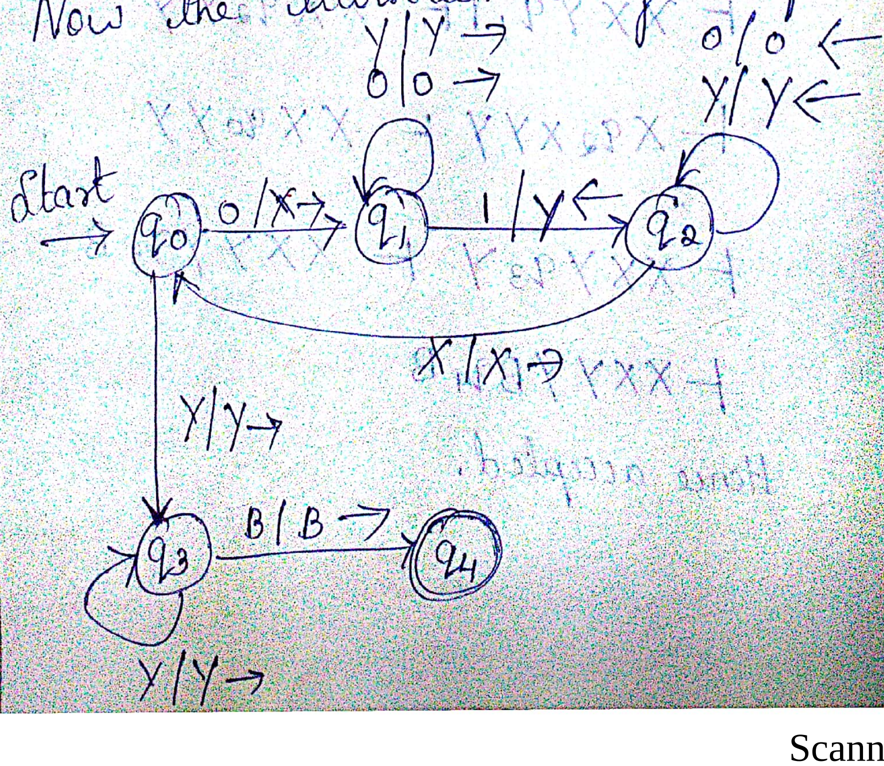
Hence accepted.

Let us check for another input string 000111

$q_0 000111 \vdash X q_1 00111 \vdash X 0 q_1 0111$
$\vdash X 0 0 q_1 111 \vdash X 0 q_2 0 Y 11$
$\vdash X q_2 0 0 Y 11 \vdash q_2 X 0 0 Y 11$
$\vdash X q_0 0 0 Y 11 \vdash X X q_1 0 Y 11$
$\vdash X X 0 q_1 Y 11 \vdash X X 0 Y q_1 11$
$\vdash X X 0 q_2 Y Y 1 \vdash X X q_2 0 Y Y 1$
$\vdash X q_2 X 0 Y Y 1 \vdash X X q_0 0 Y Y 1$
$\vdash X X X q_1 Y Y 1 \vdash X X X Y q_1 Y 1$
$\vdash X X X Y Y q_1 1 \vdash X X X Y q_2 Y Y$
$\vdash X X X q_2 Y Y Y \vdash X X q_2 X Y Y Y$
$\vdash X X X q_0 Y Y Y \vdash X X X Y q_3 Y Y$
$\vdash X X X Y Y q_3 Y \vdash X X X Y Y Y q_3 B$

Hence accepted.

Now the transition diagram for $0^n 1^n$



Pblm:-

Design a Turing machine to recognise all strings consisting of odd number of 1's.

Sln:- The Turing machine moves are written into a table called transition table.

$\Sigma = \{1\}$.

Tape symbol $= \{X\}$

q_0 is initial state, reading symbol (1) enter into new state q_1 and replace 1 by X.

From q_1 if we read one more one we again enter into q_0 by changing 1 to B.

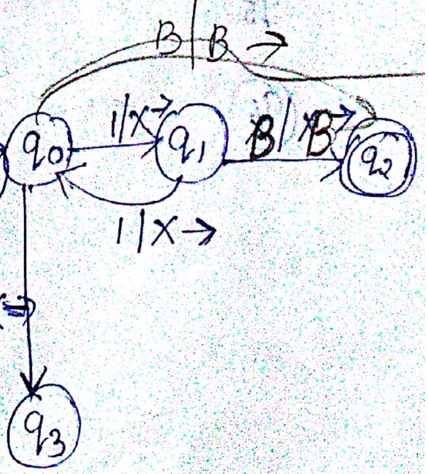
With these moves if string contains odd number of 1's then always ends with q_2 which is a final state.

If string contain even number of 1's always ends with q_0.

$M = (\{q_0, q_1, q_2, q_3\}, \{1\}, \{X\}, \delta, q_0, B, \{q_2\})$

δ is defined as

state	1	X	B
$\rightarrow q_0$	(q_1, X, R)	(q_3, X, R)	(q_2, B, R)
q_1	(q_0, X, R)	(q_2, X, R)	
q_2	-	-	
q_3	-	-	



eg: string 111

$q_0 111 \vdash X q_1 11$

$\vdash XX q_0 1 \vdash XXX q_1 B \vdash XXXB q_2$

accept

Pblm: 3

Design a turing machine to recognize the language $\{1^n 2^n 3^n \mid n \geq 1\}$

Sln:-

Let $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$

$\Sigma = \{1, 2, 3\}$

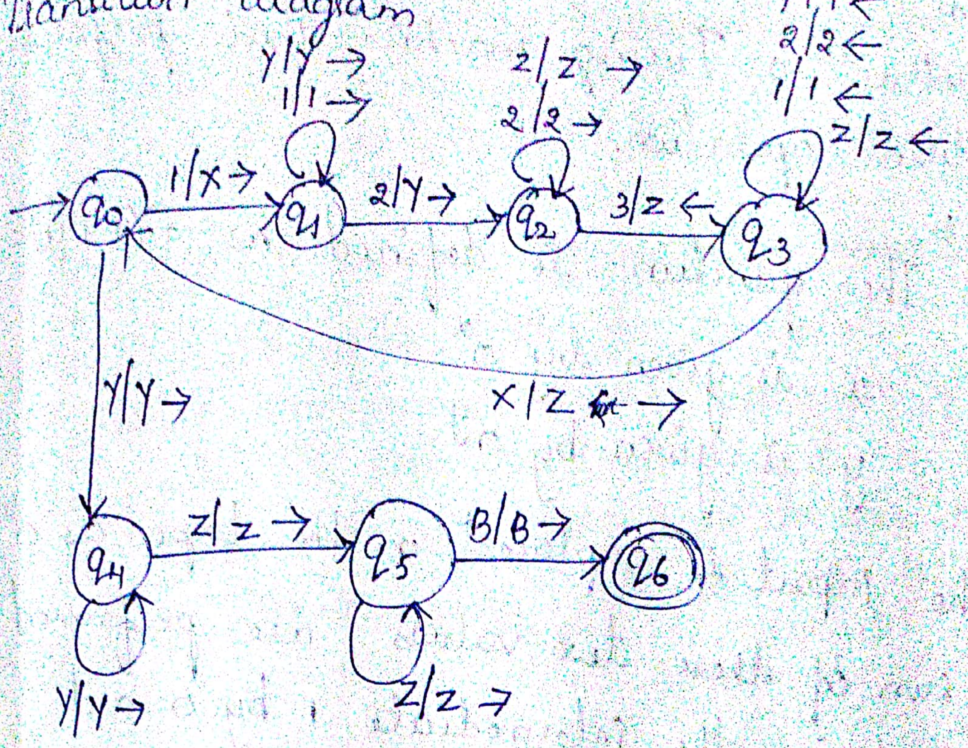
$\Gamma = \{1, 2, 3, X, Y, Z, B\}$

$F = \{q_6\}$

Transition table

state	1	2	3	X	Y	Z	B
$\rightarrow q_0$	(q_1, X, R)	–	–	–	(q_4, Y, R)	–	–
q_1	$(q_1, 1, R)$	(q_2, Y, R)	–	–	(q_1, Y, R)	–	–
q_2	–	$(q_2, 2, R)$	(q_3, Z, L)	–	–	(q_2, Z, R)	–
q_3	$(q_3, 1, L)$	$(q_3, 2, L)$	–	(q_0, Z, R)	(q_3, Y, L)	(q_3, Z, L)	–
q_4	–	–	–	–	(q_4, Y, R)	(q_5, Z, R)	–
q_5	–	–	–	–	–	(q_5, Z, R)	(q_6, B, R)
q_6	–	–	–	–	–	–	–

Transition diagram



Considering the string 123

$q_0 123 \vdash X q_1 23 \vdash XY q_2 3 \vdash XY q_3 Z$

$\vdash X q_3 YZ \vdash q_3 XYZ \vdash Z q_0 YZ$

$\vdash ZY q_4 Z \vdash ZYZ q_5 B \vdash ZYZB q_6$

Hence accepted

Consider the string 112233

$q_0 112233 \vdash X q_1 12233 \vdash X1 q_1 2233 \vdash X1Y q_2 233$

$\vdash X1Y2 q_2 33 \vdash X1Y q_3 2Z3 \vdash X1 q_3 Y2Z3$

$\vdash X q_3 1Y2Z3 \vdash q_3 X1Y2Z3 \vdash Z q_0 1Y2Z3$

$\vdash ZX q_1 Y2Z3 \vdash ZXY q_1 2Z3 \vdash ZXYY q_2 Z3$

$\vdash ZXYYZ q_2 3 \vdash ZXYY q_3 ZZ \vdash ZXY q_3 YZZ$

$\vdash ZX q_3 YYZZ \vdash Z q_3 XYYZZ \vdash ZZ q_0 YYZZ$

$\vdash ZZY q_4 YZZ \vdash ZZYY q_4 ZZ \vdash ZZYYZ q_5 Z$

$\vdash ZZYYZZ q_5 B \vdash ZZYYZZB q_6$

Hence accepted.

Pblm:-

Construct a Turing machine that performs addition operation.

Sln:-

The function is defined as $f(x+y) = x+y$

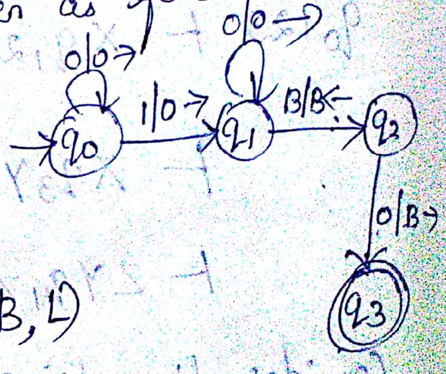
x is given by 0^x

y is given by 0^y

The input is replaced on the tape as $0^x 1 0^y$.

The sum of these two values are performed by replacing the intermediate 1 by 0 and replacing the last 0 by blank symbol.

The transition table is given as follows:



δ	0	1	B
→ q_0	$(q_0, 0, R)$	$(q_1, 0, R)$	–
q_1	$(q_1, 0, R)$	–	(q_2, B, L)
q_2	(q_3, B, R)	–	–
* q_3	–	–	–

Eg: $x = 3,\ y = 2$ $\quad f(x+y)$

(1) $q_0 0^3 1 0^2 \Rightarrow q_0 000100 \vdash 0q_0 00100$

$\vdash 00q_0 0100 \vdash 000q_0 100$

$\vdash 0000q_1 00 \vdash 00000q_1 0$

$\vdash 000000q_1 B \vdash 00000q_2 0B$

$\vdash 00000Bq_3 B$ Hence accepted

Pbm:
Design a turing machine to compute proper subtraction
(ii) $m-n$ for $m \geq n$
0 for $m < n$.

Ans:
The turing machine started its operation with $0^m 1 0^n$ on its tape. At q_0, it replaces the leading 0 by blank and search right looking for first 1. after finding it, TM searches right for 0, and change it to 1. Then move the tape head to the left till reaches the blank symbol and then enter q_0 to repeat the cycle.

0 0 1 0
B 1 1
BB

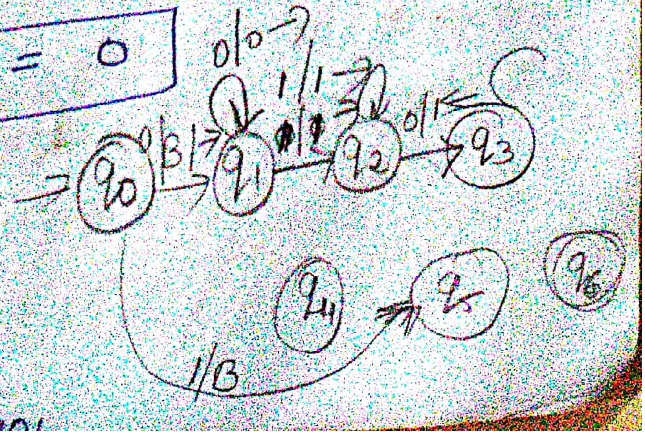
δ:

	0	1	B
$\rightarrow q_0$	(q_1, B, R)	(q_5, B, R)	-
q_1	$(q_1, 0, R)$	$(q_2, 1, R)$	-
q_2	$(q_3, 1, L)$	$(q_2, 1, R)$	(q_4, B, L)
q_3	$(q_3, 0, L)$	$(q_3, 1, L)$	(q_0, B, R)
q_4	$(q_4, 0, L)$	(q_4, B, L)	$(q_6, 0, R)$
q_5	(q_5, B, R)	(q_5, B, R)	(q_6, B, R)
$*q_6$	-	-	-

Eg: 0010

$q_0 0010 \vdash Bq_1 010 \vdash B0q_1 10 \vdash B01q_2 0 \vdash B0q_3 11$
$\vdash Bq_3 011 \vdash q_3 B011 \vdash Bq_0 011 \vdash BBq_1 11$
$\vdash BB1q_2 1 \vdash BB11q_2 B \vdash BB1q_4 1 \vdash BBq_4 1$
$\vdash Bq_4 \vdash B0q_6$

(ii) $\boxed{00 - 0 = 0}$



Pblm :-

X. Design TM to compute $f(m, n) = m * n \;\forall m, n \in N$

Sln :- let input be $0^m 1 0^n$.

Scanned by CamScanner

Programming Techniques for Turing machine construction :-

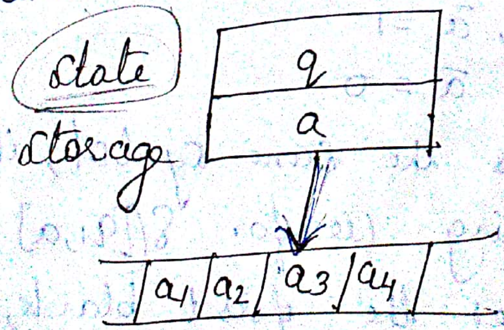
The following are the different techniques of constructing a TM to meet high level needs :

1) Storage in the finite control (or) state
2) Multiple tracks
3) Subroutines
4) Checking off symbols.

① Storage in the state (or) finite control

The finite control can also be used to hold a finite amount of information along with the task of representing of position in program.

The state is written as a pair of elements, one for control and other storing a symbol.



Ex:- Consider a Turing machine

$m = (Q, \{0,1\}, \{0,1,B\}, \delta, [q_0, B], \{[q_1, B]\}$

that looks (q_0 q_1) at the first input symbol, records in the finite control and checks that the symbol does not appear elsewhere on its input.

Sln:- Find the states of Q as $Q \times \{0, 1, B\}$

i) $= \{q_0, q_1\} \times \{0, 1, B\} = [q_0, 0], [q_0, 1], [q_0, B],$
$[q_1, 0]\ [q_1, 0], [q_1, B]$

(ii) The finite control holds both the state and the symbol

(iii) At q_0, the TM reads the first symbol a and goes to state q_1 where $a = 0$ or 1.

$\delta([q_0, B], a) = ([q_1, a], a, R)$

The second symbol is copied into the second component of the state, moves right and enters q_1.

(iv) At q_1, If the TM reads the other symbol, M skips over and moves right

$\delta([q_1, a], \bar{a}) = ([q_1, a], \bar{a}, R)$

where $\bar{a}$ is the complement of a

if $a = 0$, $\bar{a} = 1$
if $a = 1$, $\bar{a} = 0$

(v) if M reaches the same symbol, it halts without accepting (ie for $\delta([q_1, a], a)$)

vi) If M reaches the first blank, it enters the accepting state

$\delta([q_1, a], B) = ([q_1, B], B, R)$

Ex: Input $01+1$

$\Rightarrow [q_0, B]\,01+1 \vdash 0[q_1, 0]1+1 \vdash 01[q_1, 0]+1$

$\vdash 01+[q_1, 0]1 \vdash 01+1[q_1, 0]B \vdash$

$01+1B[q_1, B]$.

2) Multiple Tracks

It is also possible that a TM's input tape can be divided into several tracks. Each track can hold one symbol, and the tape alphabet of the TM consists of tuples with one components for each track.

	¢	1	0	1	1	$	B	B		
...	B	B	B	B	1	0	B	B	B	B
		B	B	1	0	1	1	B	B	

Finite Control

Example:

Design a turing machine to check whether the given input is prime or not using multiple tracks.

Solution:-

The binary input greater than 2 is placed on first track. And also the same input is placed on 3rd track. Then TM writes the number two in binary form on the second track. Then divide the third track by second as follows

The number on the 2nd track is subtracted from the third track as many times as possible, till getting the remainder. If remainder is 0, then number on first track is not prime. If the remainder is non zero, then increase the number on the second track by one. If the second track equals the first, the number given is a prime because it should be divided by one and itself.

Eg: (i) 8

	(1)	(2)	(3)	(4)	(5)
1st Track	8	8	8	8	8
2nd Track	2	2	2	2	2
3rd Track	8	6	4	2	0

Hence the given number is not prime.

Eg: (ii) 5

5	5	5
2	2	2
5	3	1

Increase the second track value by 1

5	5
3	3
5	2

Increase the second track value by 1

5	5
4	4
5	1

Increase the second track value by 1

5
5
5

Here the number on 2nd track = number on III rd track

Thus the given number is prime number

5) Checking off Symbols:- Checking off symbols is an effective way of recognising the language by TM. The input symbols are placed on input tape, the symbol which is read, is marked by any special character. The tape head, then can be moved to right or left.

4) Subroutines :-

A problem with some tasks to be repeated for repeated for many number of times, can be programmed using subroutines. A Turing machine with subroutine is a set of states that perform some useful process.

Ex:

Construct a TM, M to implement the total recursive function for multiplication 0^{mn} where m and n are two positive numbers:-

Sln:- M starts with $0^m 1 0^n$ on its tape and ends with 0^{mn}. The idea is to place 1 after $0^m 1 0^n$ and then copy the block of n 0's on to the right end m times, each time erasing one of the m 0's. Then the remaining n 0's and 1's are to be erased and finally getting the result 0^{mn}.

Here the copying operation is repeated for m times, let us consider this operation as a subroutine copy.

Subroutine Copy

δ:

	0	1	2	B
q_1	(q_2, X, R)	$(q_4, 1, L)$	-	-
q_2	$(q_2, 0, R)$	$(q_2, 1, R)$	-	$(q_3, 0, L)$
q_3	$(q_3, 0, L)$	$(q_3, 1, L)$	(q_1, X, R)	-
q_4	-	$(q_5, 1, R)$	$(q_4, 0, L)$	-

The transition function δ for the main program is given by

	0	1	2	B
q_0	(q_6, B, R)	–	–	–
q_5	$(q_7, 0, L)$	–	–	–
q_6	$(q_6, 0, R)$	$(q_1, 1, R)$	–	–
q_7	–	$(q_8, 1, L)$	–	–
q_8	$(q_9, 0, L)$	–	–	(q_{10}, B, R)
q_9	$(q_9, 0, L)$	–	–	(q_0, B, R)
q_{10}	–	(q_{11}, B, R)	–	–
q_{11}	(q_{11}, B, R)	(q_{12}, B, R)	–	–
$*q_{12}$	–	–	–	–

Multitape turing machine

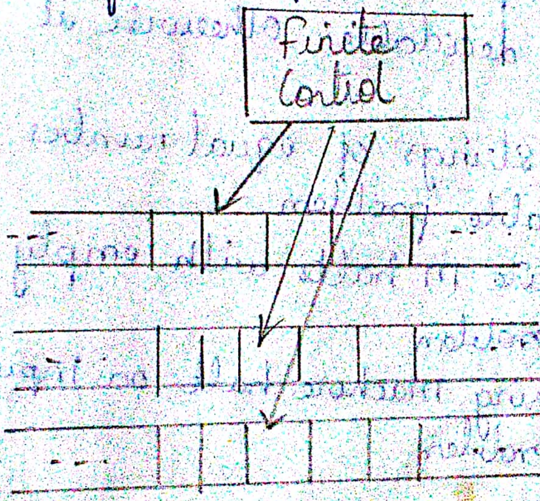
A multitape turing machine has a finite control with some finite number of tapes. Each tape is infinite in both directions. It has its own initial state and some accepting states.

Initially,

1) the finite set of input symbols is placed on the first tape.

2) All the other cells of the tapes hold the blank.

3) The control head of the first tape is at the left end of the input.



→ In one move, the multitape TM can

1) Change state.

2) Print a new symbol on each of the cells scanned by its tape heads.

3) Move each of its tape heads, independently, one cell to the left or right or keep it stationary.

Non-Deterministic Turing Machine

A NDTM is a device with a finite control and a single one-way infinite tape. For a given state and a tape symbol scanned by the tape head, the machine has a finite number of choice for the next move. For $\delta(q, X)$, there is a set of triples like

$\{(q_1, Y, D_1), (q_2, Y_2, D_2) \ldots (q_k, Y_k, D_k)\}$
where k is integer.