# Unit-5

## TURING MACHINE

1. Explain the types of turing Machine?

A) **Types of Turining Machine (or) Variant of Turining Machine (or) Modification of Turining Machine:-**

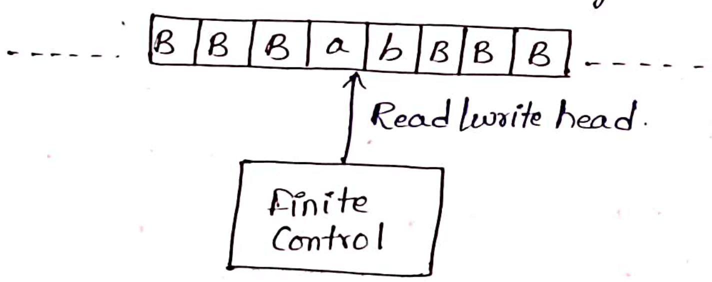
They are 7 types of Turining Machine.

① **Two-way Infinite tape Turining Machine:-**

→ The input tape in infinite at left handside and right-hand side.

→ We have infinite no.of blank Symbols at the left hand side and right hand side.

→ We Can store infinite no.of blank Symbols.

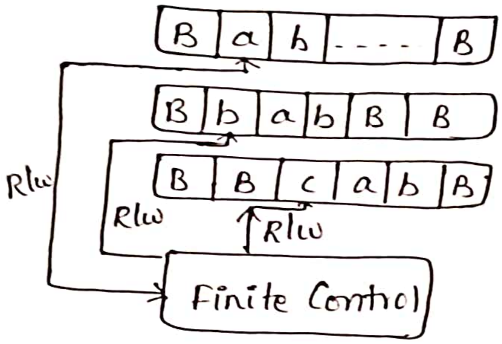



② **Multi-tape Turining Machine:-**

→ It Consists of multiple input tapes and it having infinite Size, with Single Finite Control and multiple read/write heads.

→ We Can move the read/write head either left to right (or) right to left.

→



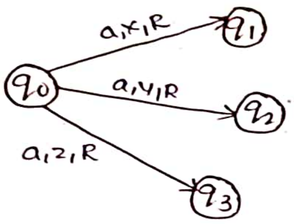

Ex:- $(q_0, a, b, c) =$
$(q_1, x, y, z, R, L, R)$

## ③ Non-Deterministic Turining Machine:-

→ By applying input Symbol you can go to multiple transitions.

→ Instead of One choice it have multiple choices.

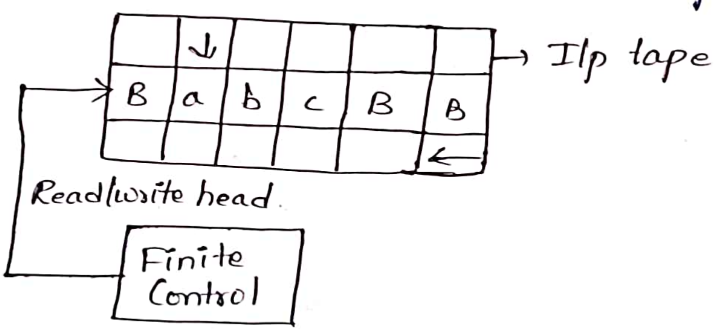
Ex:-




## ④ Multi- Dimensional Turining Machine:-

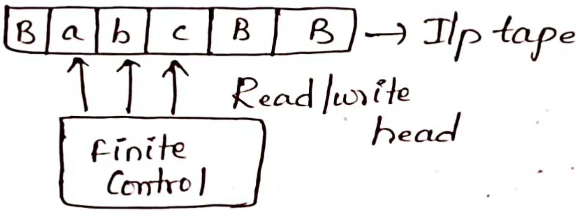
→ Instead of One-dimension we have multiple dimensions in input tape with One finite Control and one read/write head.

→ We Can move Read/write head moves towards either left to righ (or) right to left (or) up to down (or) down to up



## Multihead Turining Machine:-

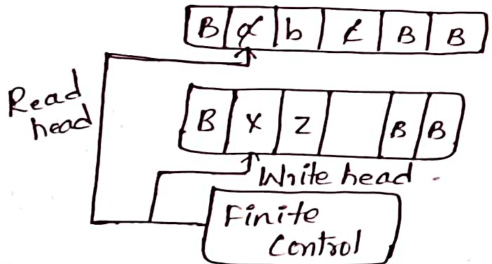
→ Instead of one-read/write head we can have multiple read/write heads with One-finite Control and One input tape.




Ex:- $(q_0, a, b, c) = (q_1, X, Y, Z, R, R, C)$

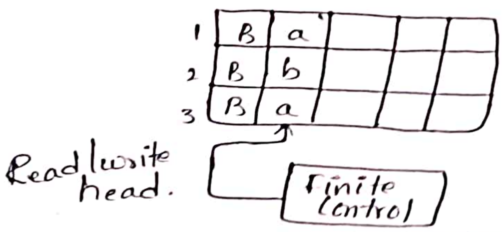
⑥ ## OFFline Turining Machine:-

→ It Consists of multiple input tapes with One finite Control, but here read Operation performs in one input tape and Coresponding write Operation performs in another input tape.




⑦ ## Multi track Turining Machine:-

→ It Consist only One input tape, One finite Control and One read/write head, but input tape is divided into multiple tracks.



Ex:- $(q_0, a, b, a) = (q_1, x, y, x, R, R, R)$

2. Define the Turining Machine with formal Notations. Explain the Concept of Universal Turing Machine.

A) **Formal Notations:-**

A Turing Machine Can be defined as a Set of 7 tuples.

$$(Q, \Sigma, \Gamma, \delta, q_0, b, F)$$

$Q$ = Non Empty Set of States

$\Sigma$ = Non Empty Set of input Symbols.

$\Gamma$ = Non Empty Set of Tape Symbols.

$\delta$ → Transition function defined as

$$Q \times \Sigma = \Gamma \times (R|L) \times Q$$

$q_0$ → Initial State

$b$ → Blank Symbol

$F$ → Set of Final States (Accept State & Reject State)

Thus, the production rule of Turing Machine will be written as:

$$\delta(q_0, a) \longrightarrow (q_1, Y, R)$$

**Universal Turing Machine:-** (U.T.M)

→ A Turing Machine is Said to be Universal Turing Machine, if it Can Simulate the behaviour of any Turing Machine.

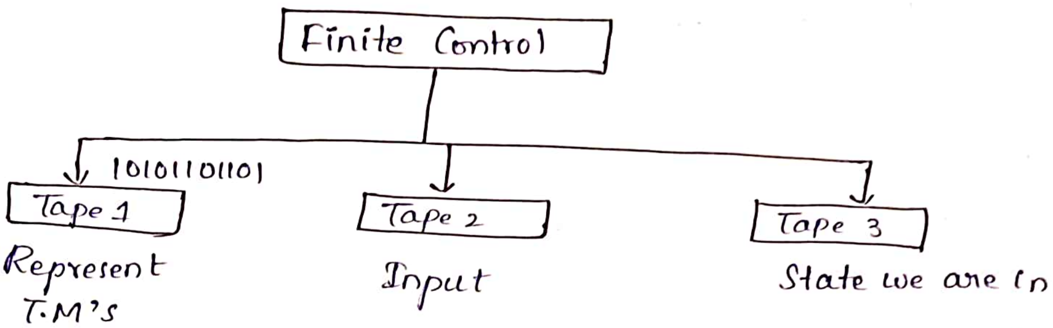
→ A Standard Turing Machine Can Simulate only a Single Computa-tion problem. So, if we want to Solve another Computation problem then another T.M is retund to be Constructed.

→ In Case of U.T.M, Some UTM to be Used to Solve any Computa-tion problem.

→ Standard T.M is Un programmable T.M as it works only for One Computation problem. However Universal T.M is "programmable T.M" as it works for all Solvable Computation problem.

→ power of U.T.M is Same as power of a standard T.M as both Solve the Same Set of Computational problem.

Block diagram:-




→ It Consists of F.C, Tape 1, Tape 2 and Tape 3

→ Tape 1 is Used to representation of T.M's Suppose we you are having a T.M for addition and place on a Tape 1, like Subtraction also.

→ Tape 2 is Used to represent i/p i.e i/p's are placed in tape 2.

→ Tape 3 is representation which state we are in i.e either in $q_0 | q_1 | q_2$ ....

Representation of Turing Machine:-

Ex:-

| | | | | | | |
|---|---|---|---|---|---|---|
| $q_0$ | $q_1$ | $q_2$ | a | b | c | R |
| 1 | 11 | 111 | 1 | 11 | 1 | 11 |

$\delta(q_0, a) = (a, b, L)$

1 0 1 0 1 1 0 1 1 0 1

↓

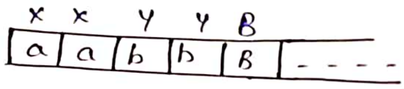
This can be placed in a tape 1

→ I/p to U.T.M is binary in Coding of Turing Machine and String "w".

3. Design Turing machine and its transition diagram to accept the language $L = \{a^n b^n \mid n >= 1\}$

A) $L = \{ab, aabb, aaabbb, ....\}$

Algorithm:-




→ change "a" to "x"

→ Move RIGHT to first "b"

If None: REJECT

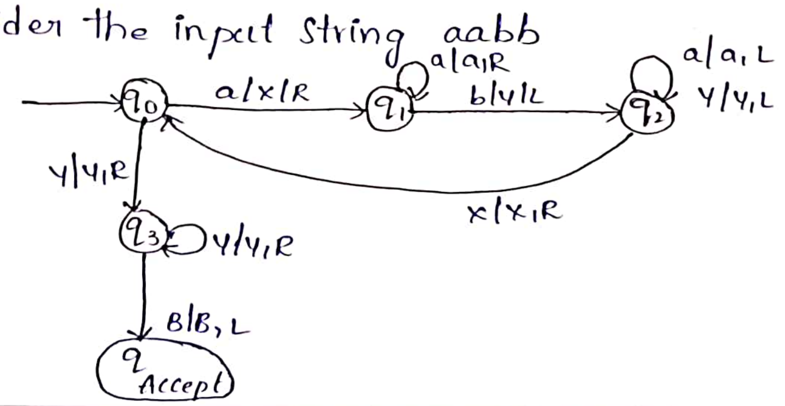
→ Change "b" to "y"

→ Move LEFT to leftmost "a"

→ Repeat the above steps Until no more "a"s

→ Make Sure no more "b"s remain

let Consider the input String aabb

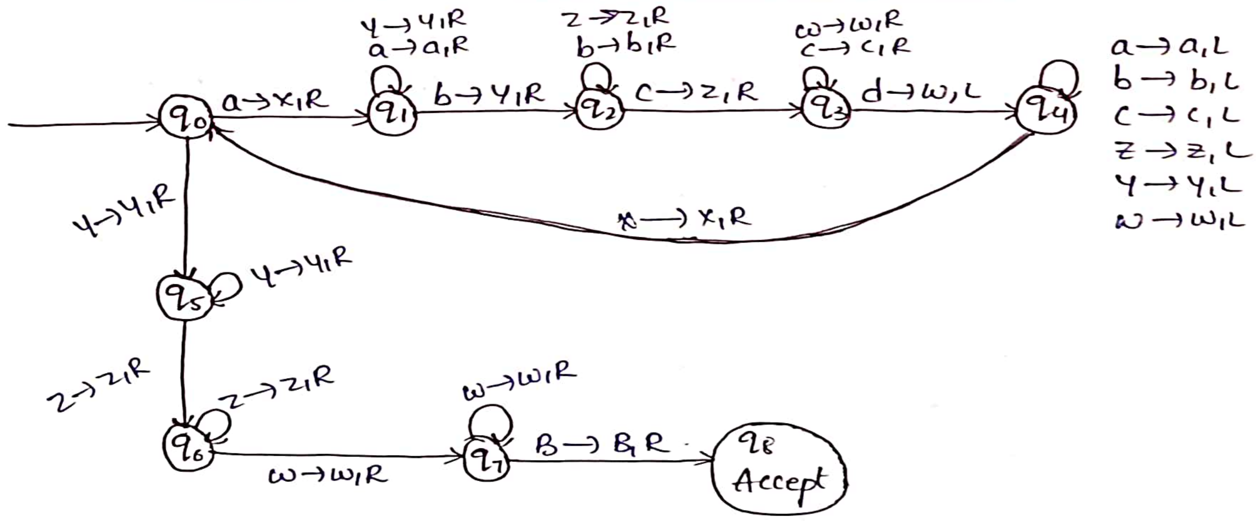


| Tape Symbol / State | a | b | X | Y | B |
|---|---|---|---|---|---|
| → $q_0$ | $\langle q_1, X, R\rangle$ | — | — | $\langle q_3, Y, R\rangle$ | — |
| $q_1$ | $\langle q_1, a, R\rangle$ | $\langle q_2, Y, L\rangle$ | — | $\langle q_1, Y, R\rangle$ | — |
| $q_2$ | $\langle q_2, a, L\rangle$ | — | $\langle q_0, X, R\rangle$ | $\langle q_2, Y, L\rangle$ | — |
| $q_3$ | — | — | — | $\langle q_3, Y, R\rangle$ | $\langle q_A, B, L\rangle$ |
| * $q_{Accept}$ | — | — | — | — | — |

4. Design a Turing Machine to accept the language $L = \{a^n b^n c^n d^n / n >= 1\}$

A) $L = \{abcd, aabbccdd, aaabbbcccddd, \ldots\ldots\}$ → Consider the i/p string aaabbbcccddd

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | X | X | X | Y | Y | Y | Z | Z | Z | W | W | W | |
| B | ~~a~~ | ~~a~~ | ~~a~~ | ~~b~~ | ~~b~~ | ~~b~~ | ~~c~~ | ~~c~~ | ~~c~~ | ~~d~~ | ~~d~~ | ~~d~~ | B |




| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| B | X | X | X | Y | Y | Y | Z | Z | Z | W | W | W | B |

| Tape Symbol / State | a | b | c | d | X | Y | Z | W | B |
|---|---|---|---|---|---|---|---|---|---|
| $q_0$ | $\langle q_1, X, R\rangle$ | – | – | – | – | $\langle q_5, Y, R\rangle$ | – | – | – |
| $q_1$ | $\langle q_1, a, R\rangle$ | $\langle q_2, Y, R\rangle$ | – | – | – | $\langle q_1, Y, R\rangle$ | – | – | – |
| $q_2$ | – | $\langle q_2, b, R\rangle$ | $\langle q_3, Z, R\rangle$ | – | – | – | $\langle q_2, Z, R\rangle$ | – | – |
| $q_3$ | – | – | $\langle q_3, c, R\rangle$ | $\langle q_4, W, L\rangle$ | – | – | – | $\langle q_3, W, R\rangle$ | – |
| $q_4$ | $\langle q_4, a, L\rangle$ | $\langle q_4, b, L\rangle$ | $\langle q_4, c, L\rangle$ | – | $\langle q_0, X, R\rangle$ | $\langle q_4, Y, L\rangle$ | $\langle q_4, Z, L\rangle$ | $\langle q_4, W, L\rangle$ | – |
| $q_5$ | – | – | – | – | – | $\langle q_5, Y, R\rangle$ | $\langle q_5, Z, R\rangle$ | – | – |
| $q_6$ | – | – | – | – | – | – | $\langle q_6, Z, R\rangle$ | $\langle q_6, W, R\rangle$ | – |
| $q_7$ | – | – | – | – | – | – | – | $\langle q_7, W, R\rangle$ | $\langle q_8, B, L\rangle$ |
| $q_8$ Accept | – | – | – | – | – | – | – | – | – |

5. Explain the Concepts of NP-Hard and NP-Complete with Examples.

<u>NP-hard</u>:

→ NP-Hard Problems (say X) can be solved if and only if there is a NP-Complete problem (say Y) that can be reducible into X in polynomial time.

→ To solve this problem, it do not have to be in NP.

→ Time is unknown in NP-Hard.

→ NP-Hard is not a decision problem.

→ Not all NP-hard problems are NP-Complete, Do not have to be a Decision problem.

<u>Example</u>: Halting Problem, vertex cover problem, etc.

For Halting problem refer 7th question.

## NP-Complete:

- NP-complete problems can be solved by a non-deterministic algorithm/ Turing Machine in polynomial time.
- To solve this problem, it must be both NP and NP-hard problems.
- Time is known as it is fixed in NP-Hard.
- NP-Complete is exclusively a decision problem.
- All NP-Complete problems are NP-hard.
- It is exclusively a Decision problem.

Example:

1. Travelling Salesman Problem (TSP):

   Given a list of cities and the distances between each pair of cities, find the shortest possible route that visits each city exactly once and returns to the origin city.

2. Graph Coloring Problem: Given an undirected graph, assign colours to vertices so that no 2 adjacent vertices have the same color using the fewest possible colors.

6. Construct the Turing machine that Computes Subtraction, where the first Operand length is more than the Second Operand. X is a Symbol that Separates the two Operands.

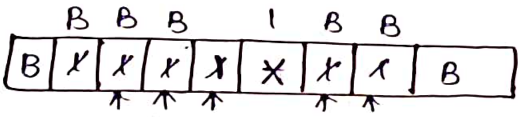
A)

$$f(m,n) = \begin{cases} m-n, & \text{if } m \geq n \\ 0, & \text{if } m < n \end{cases}$$

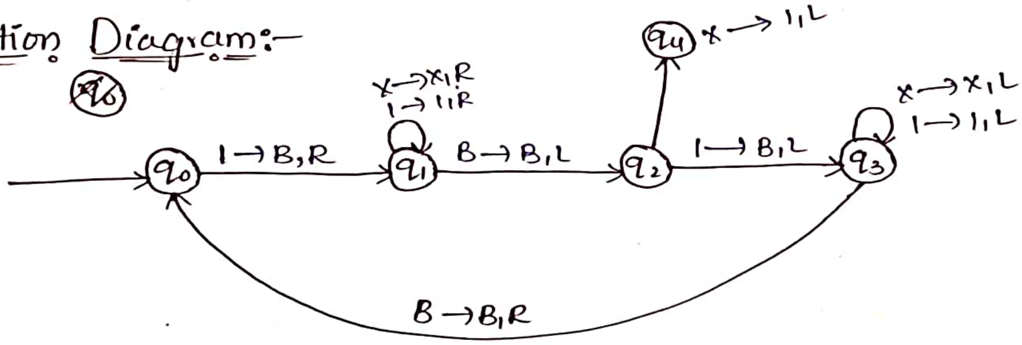
$m=4$, $n=2$

$\downarrow$ $\quad\quad$ $\downarrow$

1111 $\quad$ 11




Transition Diagram:-




| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| B | B | B | B | 1 | 1 | B | B | B |

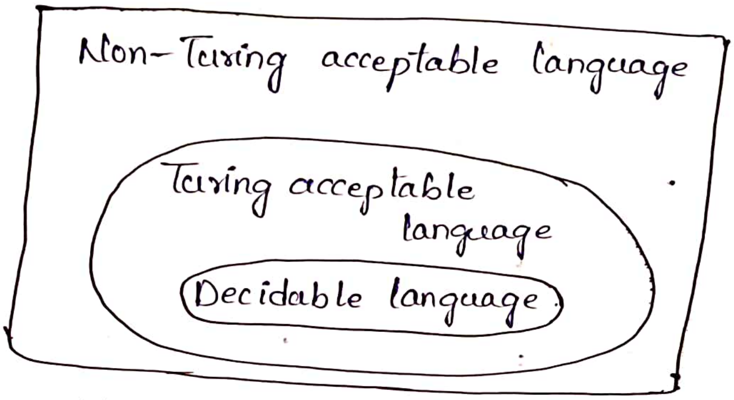
| State \ Tape symbol | 1 | X | B |
|---|---|---|---|
| $q_0$ | $\langle q_1, B, R\rangle$ | | |
| $q_1$ | $\langle q_1, 1, R\rangle$ | $\langle q_1, X, R\rangle$ | $\langle q_2, B, L\rangle$ |
| $q_2$ | $\langle q_3, B, L\rangle$ | $\langle q_4, 1, L\rangle$ | |
| $q_3$ | $\langle q_3, 1, L\rangle$ | $\langle q_3, X, L\rangle$ | $\langle q_0, B, R\rangle$ |
| $q_4$ | – | [illegible] | – |

⑥

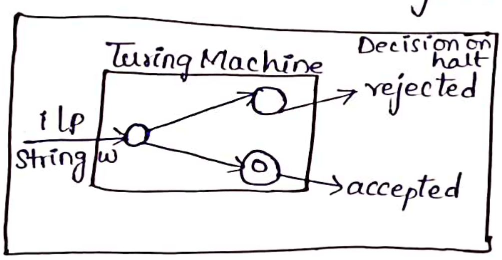
Discuss the decidable and Undecidable problems with Examples.

1) Decidable problem:-

* A language is Called Decidable (or) recursive if there is a turing machine which accepts and halts on every i/p string "w".
* Every decidable language is a turing acceptable.




* A decision problem 'p' is decidable. If the language 'L' of all yes instances to 'p' is decidable.
* for a decidable language for each i/p string, the turing machine halts either at the accept (or) the reject state.




Example:-

1. Find out whether the following problem is decidable (or) not. Is a number is 'm' prime?

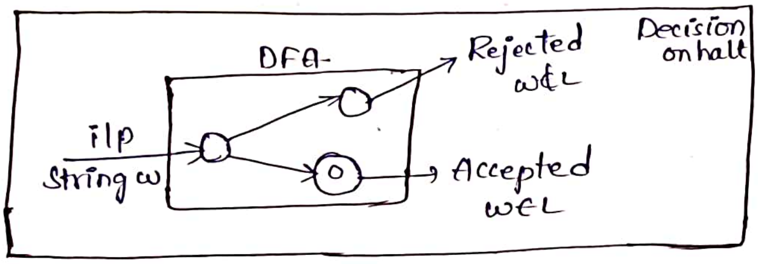
Sol:- prime numbers = $\{2,3,5,7,11,13,17,19 \ldots\}$

divide the number 'm' by all the numbers b/w 2 and m/2 Starting from 2.

If any of these numbers produce a remainder '0' then it goes to the rejected State. otherwise it goes to the accepted state. So, Here the answer could be made by yes (or) No.

Hence, it is a decidable problem.

↳ Take the DFA that accepts 'L' and check if 'w' is accepted.




Some more decision problems are

i) Does DFA accept the empty language

ii) Is $L_1 \cap L_2 = \phi$ for regular Sets.

iii) If a language L is decidable then its Complement is also decidable.

iv) If a language is decidable then there is a turing machine for it

## Undecidable problems:-

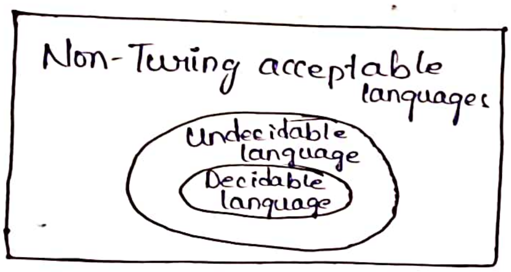
### Introduction:-

* for an Undecidable language there is no "TM" which accept the

language and makes a decision for every i/p string "w".

* A decision problem "p" is called Undecidable if the language "L" of all 'yes' instances to "p" is not decidable.

Undecidable languages are not recursive languages but sometimes they may be recursive enumerable languages.




Examples:-

i) The halting problem of turing machine.
ii) The Mortality problem.
iii) The mortal Matrix problem.
iv) The post Correspondence problem [pcp]

→ The halting problem:-

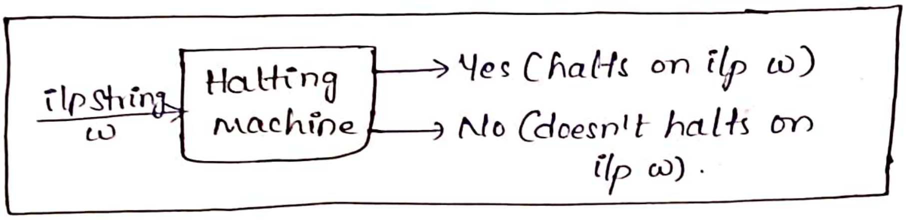
The halting problem i/p: a turing machine and the i/p string w.

Problem:- Does the turing machine finish computing of the string 'w', in a finite no. of steps? The answer must be either yes (or) No.

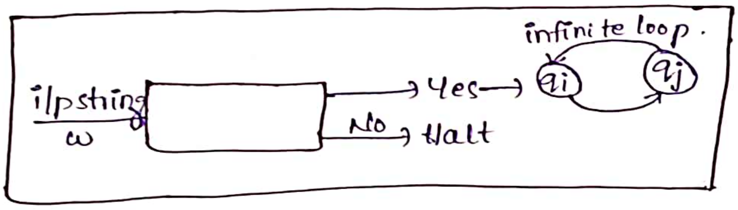
Proof:- At first, we will assume that a turing machine exists to solve, the problem. we will show and then it is contradic-ting itself.

We will Call this turing machine as a halting machine that Produces a yes (or) NO in a finite amount of time.

If the halting machine finishes in a finite amount of time then the o/p Comes as yes. otherwise, as NO




The following block diagram Shows the inverted halting machine.




further, a machine "HM" which i/p itself is Constructed as follows.

i) IF HM halts on i/p loop forever.

ii) ElSE Halt.

∴ Here, we have got a Contradiction. Hence, the halting problem is Undecidable.

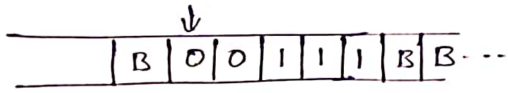
8. Design TM which accepts strings ending with 111 where the input is taken from {0,1}.

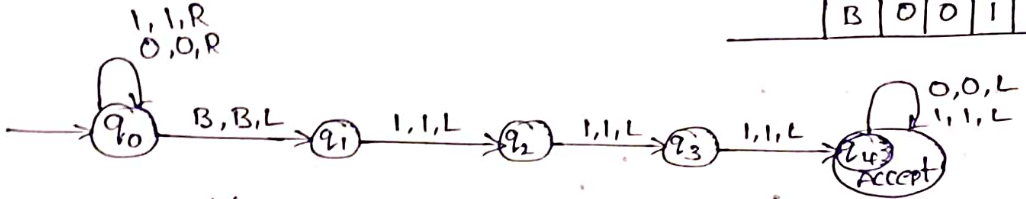
Given $\Sigma = \{0,1\}$

Condition : strings ending with 111

$L = \{ 0111, 1111, 00111, 01111, 10111, \ldots \}$

Transition Diagram:




Transition table Diagram:

| current state \ Input | 0 | 1 | B |
|---|---|---|---|
| $q_0$ | $\langle q_0, 0, R \rangle$ | $\langle q_0, 1, R \rangle$ | $\langle q_1, B, L \rangle$ |
| $q_1$ | - | $\langle q_2, 1, L \rangle$ | - |
| $q_2$ | - | $\langle q_3, 1, L \rangle$ | - |
| $q_3$ | - | $\langle q_4, 1, L \rangle$ | - |
| $q_4$ | - | - | - |

Formal Notation

$M : \{Q, \Sigma, \Gamma, \delta, q_0, B, F\}$

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

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

$\Gamma = \{0, 1, B\}$

$q_0 = \{q_0\}$

$F = \{q_4\}$

9. List the elements of TM's and give the block diagram.

Turing Machine Def :-

In Order to define turing machine by Using 7 tuples

$$M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$$

$Q$ = Finite no. of States

$\Sigma$ = input alphabet

$\Gamma$ = Finite set of Input tape Symbols.

$q_0$ = initial state

$B$ = Blank Symbol

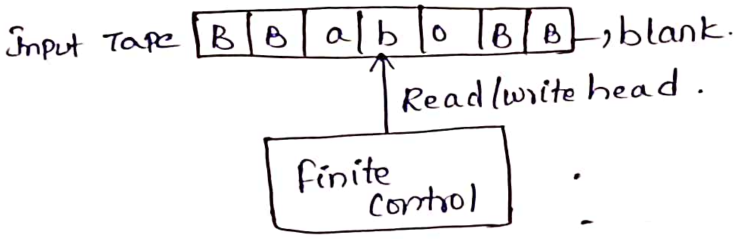
$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$

$F$ : Final state

Model of Turining Machine:-

Mainly Contains 3 Components.

1. Input tape
2. Finite Control
3. Read/write head




* Input tape is divided into no. of Celles where each Cell is Capable of Storing One Symbol at a time. The Size of the input tape is infinite. Input tape starts and ends with blank. symbol (B).
* By Using Read/write head we Can perform Read and write Operation we can Read/write 1 Symbol at a time.
* Read/write head moves from left to right (or) right to left.
* Finite Control Contains all the States.

10. Construct a TM that Computes a function $f(m,n) = m+n$ i.e, addition of two numbers.

$$f(m,n) = m+n$$

$m = 4$

$n = 2$

$f(m,n) = 4+2 = 6.$

$f(4,2) = 4+2 = 6$

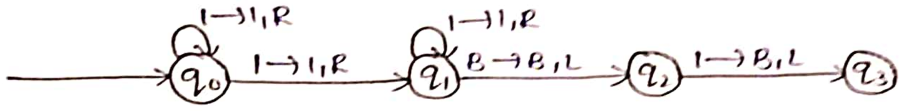
$4 \rightarrow 1111$

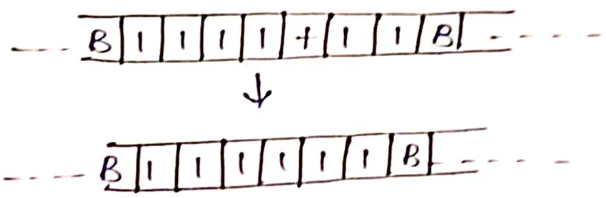
$2 \rightarrow 11$

$4+2 \rightarrow 1111+11$

$\rightarrow 111111$

## ransition diagram:-




## Transition Table:-

| Tape symbol / State | 1 | B |
|---|---|---|
| $q_0$ | $\langle q_1, 1, R \rangle$ | — |
| $q_1$ | $\langle q_1, 1, R \rangle$ | $\langle q_2, B, L \rangle$ |
| $q_2$ | $\langle q_2, B, L \rangle$ | — |
| $q_3$ | — | — |