Turing Machine

⇒ A Turing machine consist of infinitely long tape which has been divided into cells.

⇒ It contains a head, which can either move to the left or right, and can read the symbols written in the cells.

⇒ TM = PDA + input tape

TM overcome the limitation of PDA because PDA is only represent non-regular language that need to make single comparison.

⇒ where there is more than one comparison then TM will be beneficial.

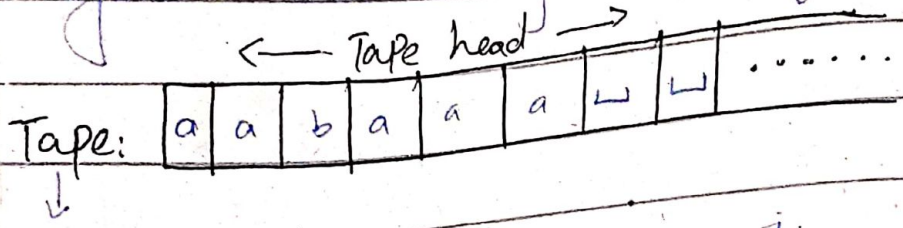
TM Tuples:

⇒ A TM can be formally described as a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, B, F)$

Q - finite number of states
B - Blank symbol
Σ - input alphabet
F - Final state
Γ - is symbols allowed on tape
q_0 - Initial state.
δ - $(Q \times \Gamma \rightarrow Q \times \Gamma \times (L, R))$

⇒ It represent recursively Enumerable language.

⇒ In Turing machine blank is a special symbol used to fill the infinite tape.



⇒ A tape is a data structure and infinite sequence of symbols

⇒ A tape head show the position where the control is present.

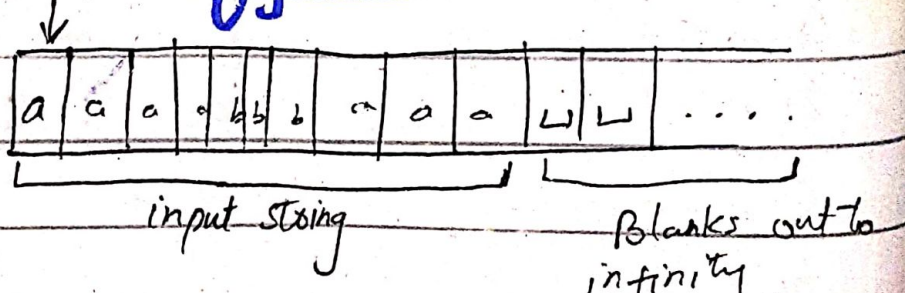
⇒ A tap head can move one head to the left and one head to the right

Tape alphabets:

⇒ $\Sigma = \{0, 1, a, b, x, z_0\}$

⇒ $\sqcup \notin \Sigma$, and Not a part of input symbol.

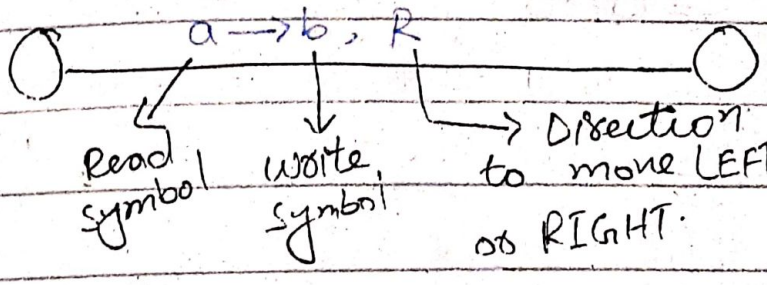
Initial configuration:



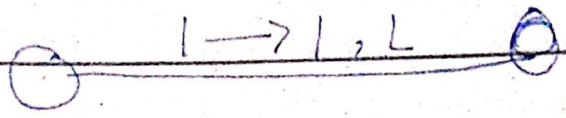
⇒ Turing machines are capable of performing any computation that can be described algorithmically.

Operations on the tape:

- Read / scan the symbol below the tape head.
- Update/write a symbol below the tape head.
- move the tape head one step LEFT.
- move the tape head one step to RIGHT.



- if don't want to update the cell just write the same symbol.



=> Initial states

=> Final states:

(1) Accept state

(2) Reject state

=> Computation can either

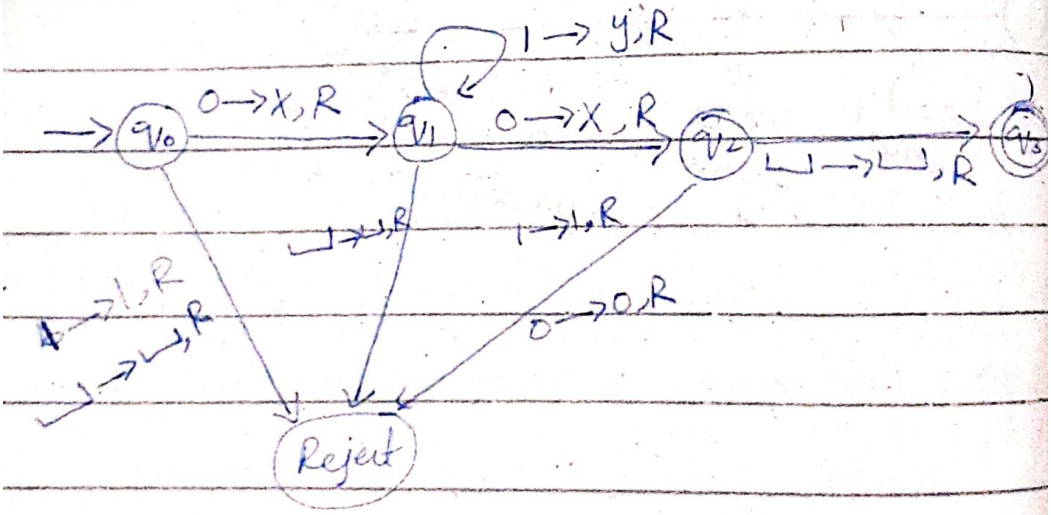
1) Halt and Accept

2) Halt and Reject

3) Loop (the machine fails to Halt)

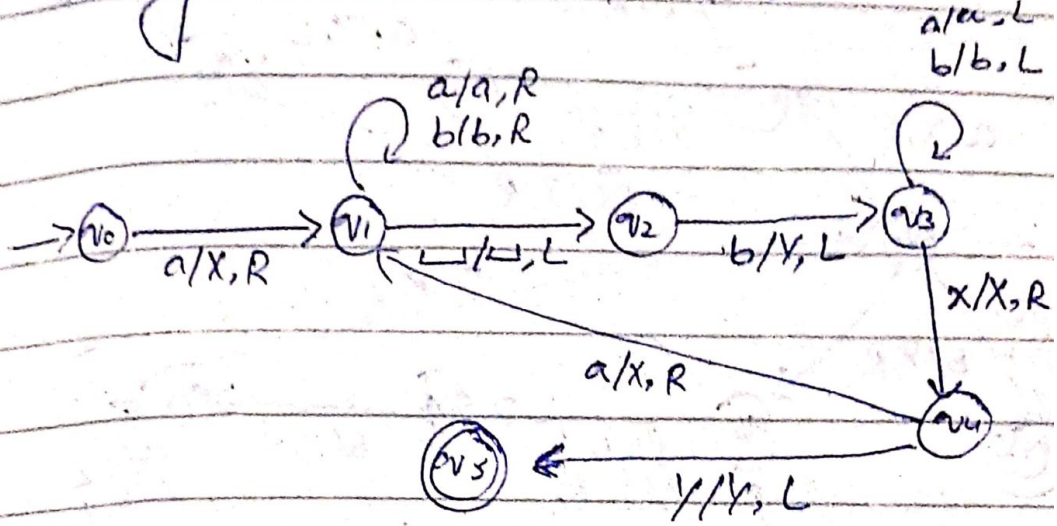
Design a turing machine that recognizer the language.

$L = 01^*0$

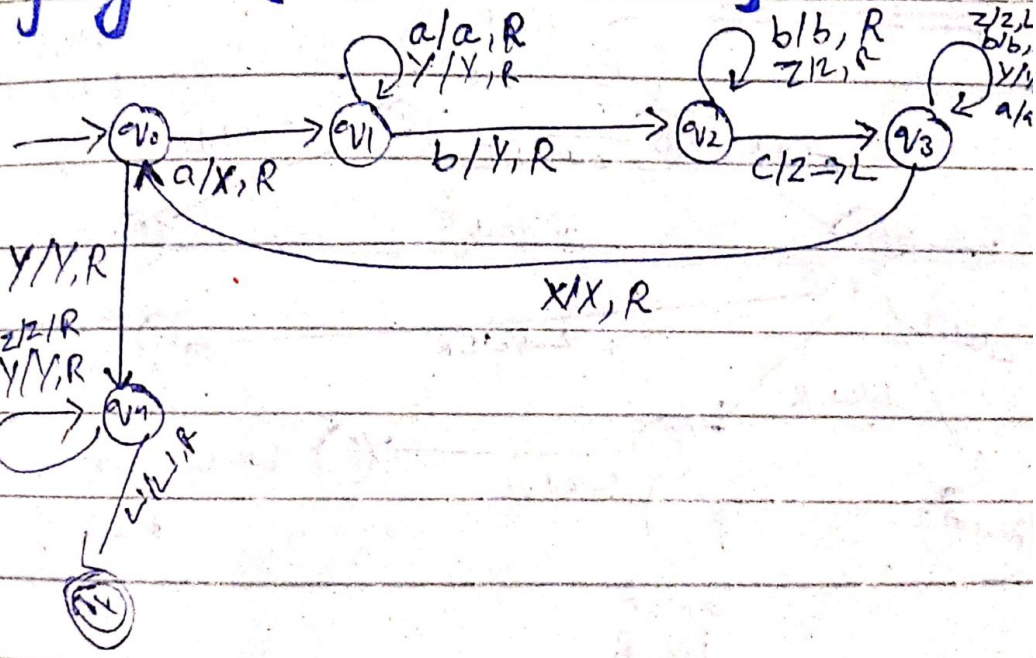


Q. Construct turing machine for language $a^n b^n$.

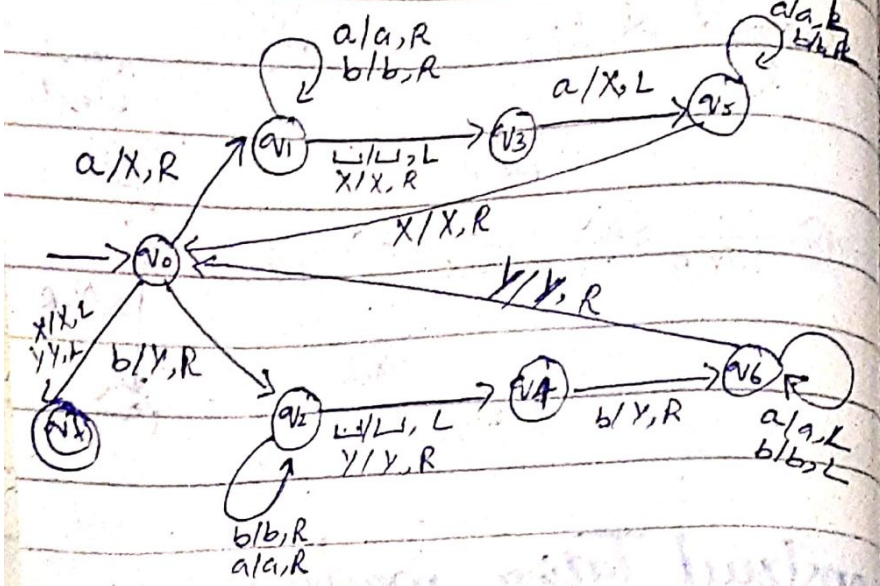
- The language need to generate by turing machine is $a^n b^n$



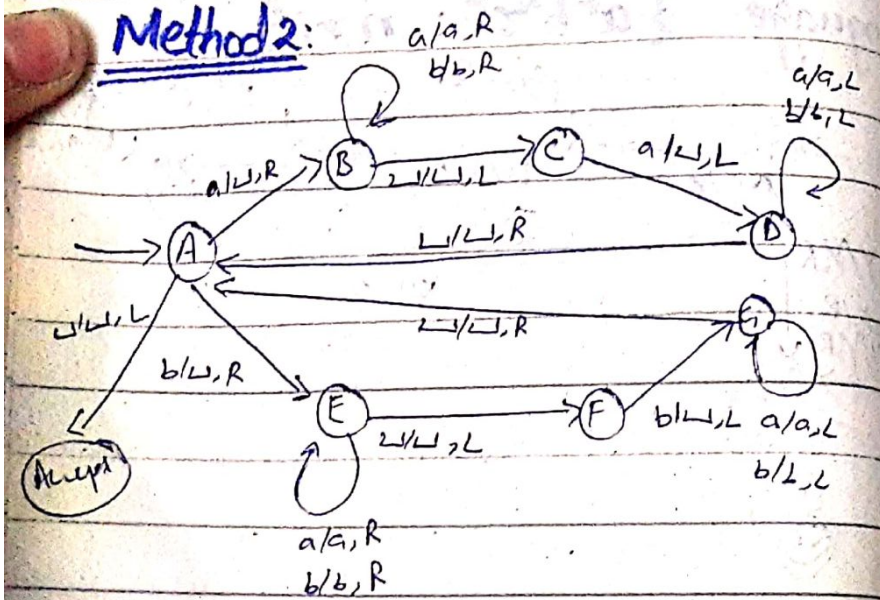
Q. Construct Turing machine for language $\{a^n b^n c^n : n \geq 1\}$



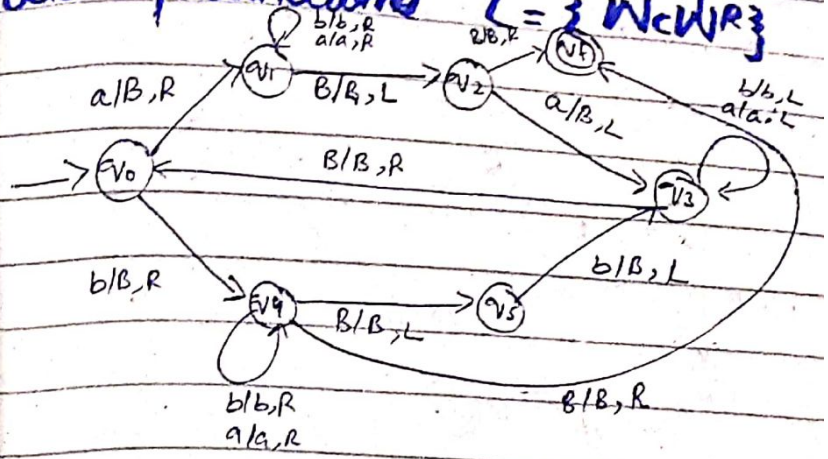
Construct Turing machine for Language WW^R



Method 2:



Construct turing machine for odd palindrome $L = \{ WcW^R \}$



⇒ The difference in odd palindrom turing machine and even palindrom is just of final state.

Past papers Question:-

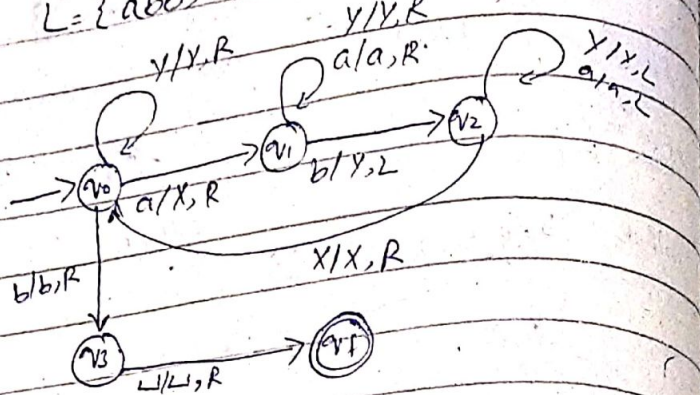
Q Construct turing machine for language.

(a) $a^n b^{n+1}$

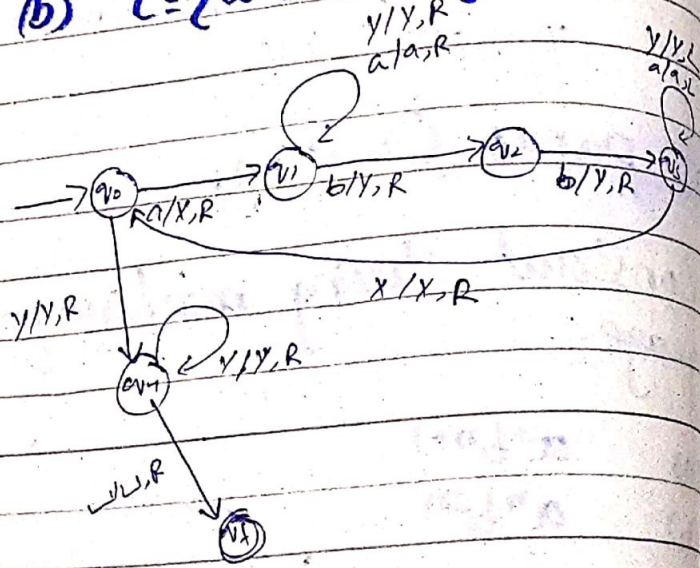
(b) $a^n b^{2n}$

1 (a) $\{a^n b^{n+1} : n \geq 1\}$

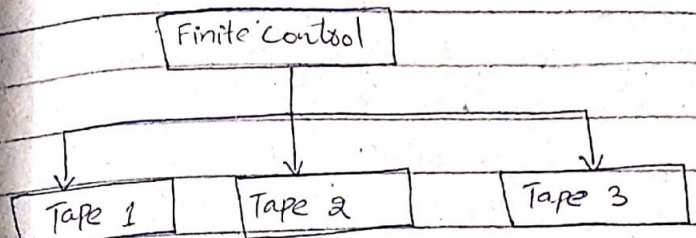
$L = \{abb, aabbb, aaabbbb \ldots\}$



(b) $L = \{a^n b^{2n} \mid n \geq 1\}$



Represent tuning machine:



represent TM's — I/P — which state we are in

What are the special feature tuning machine?

The are the following special features of tuning machine:

1) Universal Computation:-

They can simulate any algorithm or computation.

2) Tape-based Storage:

An infinite tape divided into cells for storing and manipulating symbols.

3) Read/write head:

A head that reads and writes symbols on the tape and can move left or right.

Deterministic & non-deterministic:

=> They can behave deterministically or non-deterministically.

State Transition rule:

=> They follow rules that - dep... their behavior based on the curre... state and symbol.

Halting problem:e

=> They can halt and stop comput...

Language recognition:

=> They can recognize formal lan...

Computability:-

=> They can determine whether a problem can be solved algorithmi... or if it is undecidable or unsolvabl...

=> They provide a formal model fo... understanding Computability & decidability.

Decidability & Undecidability

Recursive language:

⇒ A language 'L' is said to be recursive if there exists a Turing machine which will accept all the strings in 'L' and reject all the strings not in 'L'.

⇒ The turing machine will halt every time and give an answer (accepted or rejected) for each and every input string.

Recursive Enumerable language

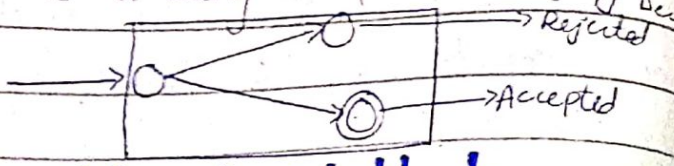
- A language 'L' is said to be a recursively enumerable language if there exists a Turing machine which will accepts (and therefore halt) for all the input strings which are in 'L'.
- But may or may not halt for all input strings which are not in 'L'

Decidable language:

⇒ A language is said to be decidable if it is a recursive language.

⇒ All decidable languages are recursive and vice versa. Every decidable language is a Turing acceptable language.

e.g



Partially decidable language:

⇒ A language 'L' is partially decidable if 'L' is a recursively enumerable language.

e.g

Undecidable language:

⇒ A language is undecidable if it is not decidable.

⇒ An undecidable language may be sometimes be partially decidable but not decidable.

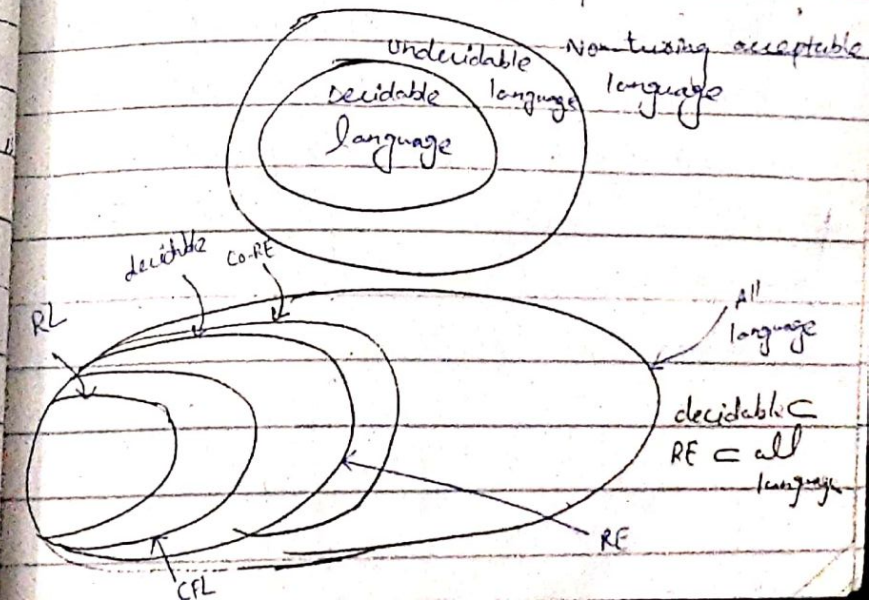
⇒ If a language is not even partially decidable, then there exist no Turing machine for that language.

⇒ The problem for which we cannot construct an algorithm that answers the problem correctly in the infinite time are termed as undecidable problems.

⇒ In other words for undecidable language there is no Turing machine that will always halt an infinite amount of time to answer as 'yes' and 'no'.

e.g

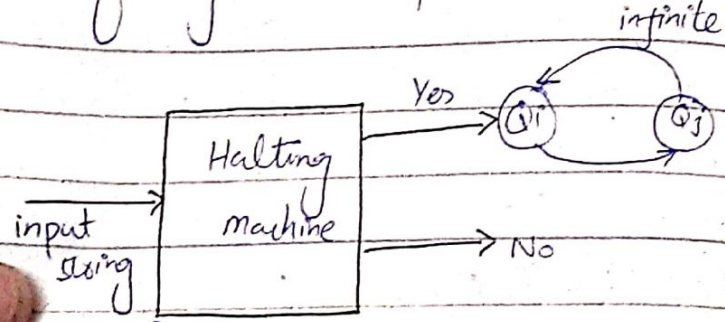
- Halting Problem
- Post Correspondance problem



What is halting problem?

⇒ Halting problem is a problem which asks whether a given Turing machine will halt onto a given input.

⇒ When a Turing machine runs a given input, it will either halt after sometime or it will keep on running infinitely on in loop.



What is PCP?

⇒ Post correspondence problem, asks whether a list of pairs of strings can be arranged in a certain way.

⇒ it was introduced by Emil Leon post in 1946.

Simplest Non halting problems:

$$\left[\begin{matrix}B\\CA\end{matrix}\right] \cdot \left[\begin{matrix}A\\AB\end{matrix}\right] \cdot \left[\begin{matrix}CA\\A\end{matrix}\right] \left[\begin{matrix}ABC\\C\end{matrix}\right]$$

Note:

⇒ Recursive language ⇒ Recursively Enumerable language	• TM will always halt • TM will halt sometimes & may not halt sometimes
⇒ Decidable language ⇒ Partially decidable language	◦ Recursive language ◦ Recursively Enumerable language
⇒ Undecidable	◦ No TM for that language.

Universal Turing machine

⇒ In universal turing machine we pass the description of another turing machine as an input and some input value.

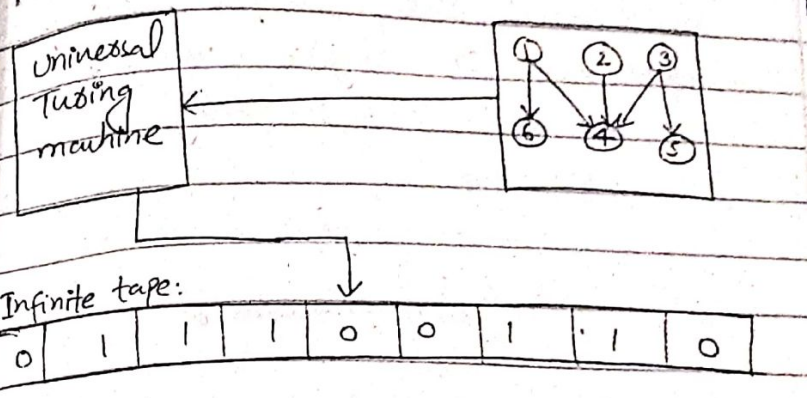
⇒ A universal turing machine takes as input a description of another turing machine, along with an input for that machine, and simulates the behavior of the specified turing machine on the given input.

⇒ And universal turing machine behaves like the turing machine whose description is provided as input.

⇒ So it may sometimes accept it may sometimes reject or sometimes may loop.

Input: M = The description of some
W = an input string of M

Action: - Simulate M.
- Behave just like M would (may accept, reject or loop).



⇒ Programmable turing machines are called universal turing machine.

⇒ Universal turing machine is a subset of all the turing machine.

⇒ Transition function:

$$Q \times T \longrightarrow Q \times T \times \{L, R\}$$

⇒ It minimizes space complexity.

⇒ A universal turing machine can compute any algorithmic computation that can be carried out by any turing machine.

⇒ The concept of universal turing machine was introduced by Alen - turing in 1936.

⇒ It can solve any algorithmic problem but some not such as halting

In universal turing machine, turing machine Act as digital computer.
we can do reprogramability.
3-Tapes are used.

Linear bound Automata

FA < PDA < LBA < TM

Regular language | CFL | context sensitive language | Recursive or recursive enumerable

- A universal TM U is deterministic TM.
- <M,w> is the input to U, and w is the input to M.
  - if M accept w, then U accept <M,w>
  - if M rejects w, then U rejects <M,w>
  - if M loops on w, then U loops <M,w>