**Theory of Computation**

**Introduction**

- It is mainly about what kind of things can you really computed mechanically , how fast and how much space does it take to do so .

- In this subject we are gonna design a machine/system that takes an input , evaluate/check it and after accepting/rejecting the input , it will give output in YES or NO .

Checks/evaluate input

MACHINE/SYSTEM

YES

NO

INPUT

**Layers/levels of TOC :-**

1 . FSM (Finite State Machine) – It is the simplest model of computation .

- It has very limited amount of memory

- It can perform very low level computations and calculations .

2 . CFL (Context Free Language) – It is little more powerful than FSM and can perform some more higher level of computations as compared to FSM .

- here the **language** means **set of strings** not a programming language

3 . Turning Machine – It can perform higher level of computations and calculations .

4 . Undecidable - The *problems that can’t be solved mechanically* , comes under this layer .

***Important terms :-***

Symbol – a , b , c , 0 , 1 , 2 ….

Alphabet (**∑**) – it is a **collection** of symbols , e.g {a , b} , {1 , 2 , 3, 4} , {0,1}

Strings – A **sequence** of symbols , e.g a , b , c , 0 , 1 , 01 , ab , abc , 01 etc

Language – **Set** of strings , e.g , L1 = with **∑=**{0,1} , set of all strings of length 2 = {00 , 11 , 10 , 01}

L2 = set of all strings of length 3 = {000 , 011 , 010 , 001}

L3 = set of all strings which begin with 0 = {0 , 00 , 01 , 000 , 0001..} *<-infinite set*

***Power of ∑ (∑x = denotes set of all strings of length x)***

suppose our alphabet is, ∑ = {0,1}

∑0 = set of all strings of length 0 : ∑0 = {£}

***Cardinality – number of elements in a set***

***Cardinality of ∑n = 2n***

∑1 = set of all strings of length 1 : ∑1 = {0 , 1} and its Cardinality = 2

∑2 = set of all strings of length 2 : ∑2 = {00, 01 , 10 , 11} , and its Cardinality = 4

∑n = set of all strings of length n

∑\* = ∑0 ∪ ∑1 ∪ ∑2 ∪……..

∑\* = {£} ∪ {0 , 1}∪ {00, 01 , 10 , 11} ∪……….

∑\* = set of all possible strings of all lengths over the alphabet{0 , 1} <-infinite set

**Finite State Machine(FSM)**