SECTION- A

FINITE AUTOMATA AND REGULAR EXPRESSIONS:

→ INTRODUCTION

→ Basic Definitions

*→ Finite State Machines/Systems

↳ Deterministic Finite Automata [DFA]

↳ Non-Deterministic Finite Automata [NDFA]

→ Equivalence of DFA and NFA (NDFA)

IMP. → Conversion of NFA to DFA

→ Finite Automata with ϵ- Moves

Introduction to Machines :-

→ Concept of Basic Machines

→ Properties & Limitations of FSM.

→ Moore & Mealy Machines.

→ Equivalence of Moore & Mealy Machines (m/c)

Introduction :-

→ Computation Problems :-

Theoretically, in Computer Science; a computational problem is a mathematical object representing a collection of questions that computers might want to solve.

For example; the problem of factoring "Given a +ve integer n, find a non-trivial Prime factor of n." is a computational problem.

↳ The field of algorithms studies methods of solving computational Problems efficiently.

↳ The complementary field of computational complexity attempts to explain why certain computational problems are intracable for computers.

⇒ Types of computational Problems :

→ Decision Problem : is a Computational Problem where the answer for every instance is either Yes or No.

eg. "Given a +ve integer n, determine if n is Prime".

eg. $L = \{2, 3, 5, 7, 11, ----\}$.

→ Search Problem :- the answers can be arbitrary strings.

for eg- factoring is a search problem where the instances are +ve integers and the solutions are collections of Primes.

Represented as Relation as all instance-solution pairs.

$$R = \{(4,2), (6,2), (6,3), (8,2), (8,4), (9,3), ----\}.$$

which consists of all Pairs of numbers (n, p) where p - is a nontrivial Prime factor of n.

→ **Counting Problem:** asks for no. of solutions to a given search problem. For eg. the counting problem associated with Primality is "Given a +ve integer n, count No. of Non-trivial Prime factors of n."

→ **Optimization Problem:-** for finding the "best possible" solution among set of all possible solutions to a search problem.
eg. – Maximum independent set Problem: "Given a graph G, find an independent set of G of maximum size".

Computational Models:

In computability Theory and computational Complexity theory a **model of Computation** is the definition of set of allowable operations used in computation and their respective costs.

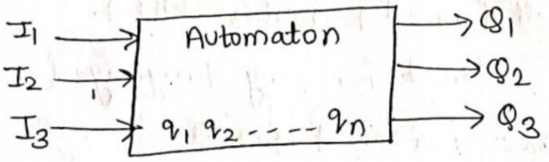
↳ used for measuring complexity of an algorithm in 'execution time' or 'memory space': by assuming a certain model of computation, it is possible to analyze the computational resources required or to discuss the limitations of algorithms or computers.

Some examples of Models include –
Turing M/cs, Recursive functions, Lambda Calculus, & Production Systems.

Automaton :-

"Automata Theory is an abstract model of computation".

- It is used for solving the computational problems.
- "Automata Theory is the study of self-operating Virtual-Machine to help in logical understanding of input and output process with or without the intermediate stage of computation".



↳ An Automaton gets one input every time step that is picked up from a set of symbols or letters which is called an 'Alphabet' and finite sequence of symbols is called 'Word' at each instance of time of run of input word. The automaton is in one of its state.

↳ At each time step, when the automaton reals the symbols of input word one after another and transitions from state to state according to transition function.

↳ Once the input words has been read the automation is said to have stopped and state at which automata has stopped is called "final State".

↳ Depending on final state, it's said that automation either accepts / Rejects an input word.
The set of all words which is accepted by automata called "language recognized by the automation".

⇒ **Language** :- A language is a set of words; i.e. finite strings of letters, symbols or tokens.

↳ The set from which these letters are taken is called 'alphabet' over which language is defined.

↳ A formal language is often defined by means of a "formal Grammer" (formation Rules).

↳ Words that belong to a formal language are sometimes called "well-formed formulas" (WFF).

→ A Hierarchy is defined for any language (or formal language)

Alphabet → a set of symbols

$$\boxed{\{a, b\}}$$

Sentences – are strings of symbols.

$$\boxed{a, b, aa, ab, ba, abb, \ldots}$$

Language – is a set of sentences

$$\boxed{L = \{aaa, aab, abaa, bbb\}}$$

Grammer – is a finite list of rules defining a language.

$S \to aA$	$B \to bB$
$A \to bA$	$B \to aF$
$A \to aB$	$F \to \epsilon$

→ Strings: "An alphabet is a non-empty finite set of symbols. denoted by Σ.

eg. $\Sigma = \{a, b, c\}$ is an alphabet.

↳ A string is a finite sequence of symbols. (U or ω)

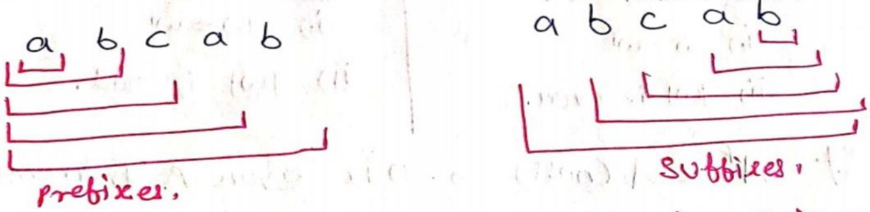
eg. $U = abcab$ is a string on $\Sigma = \{a, b, c\}$.

The empty string (no symbol at all) denoted by λ or ϵ.

– A part of string is a substring.

bca is a substring of abcab.

Note :- A beginning of a string (up to any symbol) is a prefix & an ending is a suffix.



** "A string is a prefix & suffix of itself. λ or ϵ is a prefix & suffix of any string."