UNIT – I (LANGUAGES)

Alphabets

Theory of computation is entirely based on symbols. These symbols are generally letters and digits.

Alphabets are defined as *a finite set of symbols.*

Examples:

$\sum$ = {0, 1} is an alphabet of binary digits

$\sum$ = {A, B, C, …., Z} is an alphabet.

Strings

A string is *a finite sequence of symbols selected from some alphabet*. It is generally denoted as w. For example for alphabet $\sum$ = {0, 1} w = 010101 is a string.

Length of a string is denoted as |w| and is defined as the number of positions for the symbol in the string. For the above example length is 6.

The *empty string* is the string with zero occurrence of symbols. This string is represented as ϵ or λ.

The set of strings, including the empty string, over an alphabet $\sum$ is denoted by $\sum^*$. For $\sum$ = {0, 1} we have set of strings as $\sum^*$ = {ϵ, 0, 1, 01, 10, 00, 11, 10101,…}. and $\sum 1$ = {0, 1} , $\sum 2$ = {00, 01, 10, 11} and so on.

$\sum^*$ contains an empty string ϵ. The set of non- empty string is denoted by $\sum+$. From this we get:

$\sum^* = \sum + \cup \{\epsilon\}$

Concatenation of strings

Let w1 and w2 be two strings then w1w2 denotes their concatenation w. The concatenation is formed by making a copy of w1 and followed by a copy of w2.

For example w1 = xyz, w2 = uvw
then w = w1w2 = xyzuvw
Also w13 = w1w1w1

Languages

A language is a *set of string all of which are chosen from some ∑*, where ∑ is a particular alphabet*. This means that language L is subset of ∑*. An example is English language, where the collection of legal English words is a set of strings over the alphabet that consists of all the letters. Another example is the C programming language where the alphabet is a subset of the ASCII characters and programs are subset of strings that can be formed from this alphabet.

Concatenation of Languages

If L1 and L2 are two languages then their concatenation can be defined as :

L = L1 . L2 where L = {w: w = xy where x є L1, y є L2}

It means that all the strings in the language L are concatenation of stings of language L1 and L2

Kleen Closure

If S is a set of words then by S* we mean the set of all finite strings formed by concatenating words from S, where any word may be used as often we like, and where the null string is also included.

S* is the Kleen closure for S. We can think of kleen star (S*) as an operation that makes an infinite language of strings of letters out of an alphabet.

For example for ∑ = {a}

∑* = {є, a, aa, aaa, ….}

<u>UNIT – II</u>
<u>(FINITE AUTOMATA AND REGULAR LANGUAGES)</u>

Regular languages are *languages that can be generated from one-element languages by applying certain standard operations a finite number of times*. They are the languages that can be recognized by finite automata. These simple operations include concatenation, union and kleen closure.

Regular expressions can be thought of as *the algebraic description of a regular language*. Regular expression can be *defined by the following rules*:

1. Every letter of the alphabet ∑ is a regular expression.
2. Null string є and empty set Φ are regular expressions.
3. If r1 and r2 are regular expressions, then
   (i) r1, r2
   (ii) r1r2 (concatenation of r1r2)
   (iii) r1 + r2 (union of r1 and r2)
   (iv) r1*, r2* (kleen closure of r1 and r2)
   are also regular expressions

Grammar

A grammar **G** can be formally written as a 4-tuple (N, T, S, P) where −

- **N** or $\mathbf{V_N}$ is a set of variables or non-terminal symbols.
- **T** or ∑ is a set of Terminal symbols.
- **S** is a special variable called the Start symbol, S ∈ N
- **P** is Production rules for Terminals and Non-terminals. A production rule has the form α → β, where α and β are strings on V_N ∪ ∑ and least one symbol of α belongs to V_N.

Example

Grammar G1 −

({S, A, B}, {a, b}, S, {S → AB, A → a, B → b})

Here,

- **S, A,** and **B** are Non-terminal symbols;
- **a** and **b** are Terminal symbols
- **S** is the Start symbol, S ∈ N
- Productions, **P : S → AB, A → a, B → b**

Deterministic Finite Automata (DFA) – Formal Definition

- An automaton can be represented by a quintuple (5-tuple) (Q, $\sum$, δ, q_0, F), where –
  - **Q** is a finite set of states.
  - **∑** is a finite set of symbols, called the **alphabet** of the automaton.
  - **δ** is the transition function.
  - $\mathbf{q_0}$ is the initial state from where any input is processed ($q_0 \in Q$).
  - **F** is a set of final state/states of Q ($F \subseteq Q$).

Non-Deterministic Finite Automata (NFA) – Formal Definition

- An NDFA can be represented by a 5-tuple (Q, $\sum$, δ, q_0, F) where –
- **Q** is a finite set of states.
- **∑** is a finite set of symbols called the alphabets.
- **δ** is the transition function where $\delta: Q \times \sum \rightarrow 2^Q$
- $\mathbf{q_0}$ is the initial state from where any input is processed ($q_0 \in Q$).
- **F** is a set of final state/states of Q ($F \subseteq Q$).