**Introduction: Basic Terminology: Alphabet, Formal Language**

**Alphabets, Strings and Languages:**

**Languages :**

A general definition of language must cover a variety of distinct categories: natural languages, programming languages, mathematical languages, etc. The notion of natural languages like English, Hindi, etc. is familiar to us. Informally, language can be defined as a system suitable for expression of certain ideas, facts, or concepts, which includes a set of symbols and rules to manipulate these. The languages we consider for our discussion is an abstraction of natural languages. That is, our focus here is on formal languages that need precise and formal definitions. Programming languages belong to this category. We start with some basic concepts and definitions required in this regard.

We have got acquainted with the formal notion of strings that are basic elements of a language. In order to define the notion of a language in a broad spectrum, it is felt that it can be any collection of strings over an alphabet.

Thus we define a language over an alphabet Σ as a subset of Σ\*.

Example:

1. The empty set Ø is a language over any alphabet. Similarly, {∈}is also a language over any alphabet.

2. The set of all strings over {0,1}that start with 0.

3. The set of all strings over {a,b,c}having *ac* as a substring.

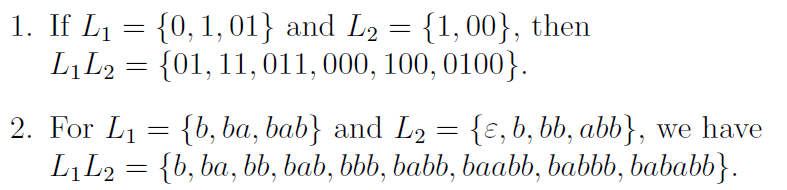
*Remark:* Note that Ø ≠ {∈}, because the language Ø does not contain any string but {∈}contains a string, namely *"*. Also it is evident that │Ø│= 0 ; whereas, │{∈}│=1

Since languages are sets, we can apply various well known set operations such as union, intersection, complement, difference on languages. The notion of concatenation of strings can be extended to languages as follows.

The concatenation of a pair of languages *L*1, *L*2 is

*L*1 *L*2 = { xy │ x ∈ *L*1 ʌ y ∈ *L*2 }.

Example



Remarks



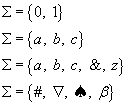
**Symbols :**

Symbols are indivisible objects or entity that cannot be defined. That is, symbols are the atoms of the world of languages. A symbol is any single object such as ♠, *a*, 0, 1, #, begin, or do. Usually, characters from a typical keyboard are only used as symbols.

**Alphabets :**

An alphabet is a finite, nonempty set of symbols. The alphabet of a language is normally denoted by Σ. When more than one alphabets are considered for discussion, then subscripts may be used (e.g. Σ1, Σ2 etc) or sometimes other symbol like *G* may also be introduced.

**Example :**



**Strings or Words over Alphabet :**  
  
A string or word over an alphabet https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-2.gif is a finite sequence of concatenated symbols of Σ.

**Example :** 0110, 11, 001 are three strings over the binary alphabet { 0, 1 } .

*aab*,*abcb*,*b*,*cc* are  four strings over the alphabet { *a*,*b*,*c*}.

It is not the case that a string over some alphabet should contain all the symbols from the alphabet. For example, the string cc over the alphabet { *a*,*b*,*c*} does not contain the symbols a and b. Hence, it is true that a string over an alphabet is also a string over any superset of that alphabet.

**Length of a string :**  
The number of symbols in a string w is called its length, denoted by |*w*|.

**Example :** |11| = 2,  |*b*| = 1

It is convenient to introduce a notation e for the empty string, which contains no symbols at all. The length of the empty string e is zero, i.e., |*e*| = 0.

**Convention :**  We will use small case letters towards the beginning of the English alphabet to denote symbols of an alphabet and small case letters towards the end to denote strings over an alphabet. That is, *a,b,c* ∈ Σ (symbols) and *u,v,w,x,y,z*  are strings.

**OPERATIONS:**

**Some String Operations :**Let x = a1a2a3 ∈ an and y = b1b2b3 ∈ bm  be two strings. The concatenation of *x* and y denoted by *xy*, is the string a1a2a3…… anb1b2b3……. bm. That is, the concatenation of *x* and *y* denoted by *xy* is the string that has a copy of *x* followed by a copy of *y* without any intervening space between them.

**Example :** Concatenation of the strings 0110 and 11 is 011011 and concatenation of the strings good and boy is goodboy.

Note that for any string *w*, *we*=*ew*=*w*. It is also obvious that if | *x* | = *n* and | *y* | = *m*, then | *x* + *y* | = *n* + *m*.

*u* is a prefix of *v* if *v*=*ux* for some string *x*.  
*u* is a suffix of *v* if *v*=*xu* for some string*x*.  
*u* is a substring of *v* if *v*=*xuy* for some strings *x* and *y*.

**Example :**  Consider the string 011 over the binary alphabet. All the prefixes, suffixes and substrings of this string are listed below.

Prefixes: *e*, 0, 01, 011.  
Suffixes: *e*, 1, 11, 011.  
Substrings: *e*, 0, 1, 01, 11, 011.

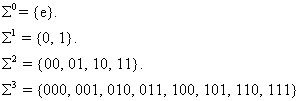
Note that *x* is a prefix (suffix or substring) to *x*, for any string *x* and *e* is a prefix (suffix or substring) to any string.

A string *x* is a proper prefix (suffix) of string *y* if *x* is a prefix (suffix) of *y* and *x* ≠ *y*.

In the above example, all prefixes except 011 are proper prefixes.

**Powers of Strings** **:** For any string x and integer n ≥ 0 , we use xn  to denote the string formed by sequentially concatenating *n* copies of *x*. We can also give an inductive definition of xn as follows:  
xn = *e*, if *n*= 0 ; otherwise xn = xxn-1

**Example :** If *x*= 011, then x3 = 011011011, x1 = 011 and x0 = *e*

**Powers of Alphabets :**We write Σk (for some integer *k*) to denote the set of strings of length *k* with symbols from Σ. In other words,  
Σk = { *w* | *w* is a string over Σ and  |*w*| = *k*}. Hence, for any alphabet, Σ0 denotes the set of all strings of length zero. That is, Σ0 = {*e* }. For the binary alphabet { 0, 1 } we have the following.  
**

The set of all strings over an alphabet Σ is denoted by  Σ\* . That is,

**https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-21.gif**

The set Σ\* contains all the strings that can be generated by iteratively concatenating symbols from Σ any number of times.

**Example :** If Σ = { *a*,*b*}, then Σ\* =  {*e*,*a*,*b*,*aa*,*ab*,*ba*,*bb*,*aaa*,*aab*, *aba*,*abb*,*baa*,*…*}.

Please note that if Σ= F, then  Σ\* that is Ø\* = {*e* }. It may look odd that one can proceed from the empty set to a non-empty set by iterated concatenation. But there is a reason for this and we accept this convention.

The set of all nonempty strings over an alphabet Σ is denoted by  Σ\*. That is,

https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-24.gifNote that Σ\*is infinite. It contains no infinite strings but strings of arbitrary lengths.

**Reversal :**For any string w = a1 a2 a3 ……an  the reversal of the string is wR = an an-1 ……. a3 a2 a1 .

**RELEVANT READING MATERIAL AND REFERENCES:**

**Source Notes:**

1. <https://nptel.ac.in/courses/106/103/106103070/>
2. <https://www.iitg.ac.in/dgoswami/Flat-Notes.pdf>

**Lecture Video:**

1. <https://www.youtube.com/watch?v=iVZDqRQiPMo>

**Online Notes:**

1. <https://www.iitg.ac.in/dgoswami/Flat-Notes.pdf>

**Text Book Reading:**

1. Martin J.C., “Introduction *to Languages and Theory of Computation*”, Tata McGraw-Hill Publishing Company Limited, 3rd Edition.
2. Hopcroft J.E. and Ullman J.D., “Introduction *to Automata Theory Languages and Computation*”, Narosa Publications.