ASSIGNMENT-2

Graph:
A graph is formed by some set of points which are connected to each other through straight lines. In the graph, the points are called nodes or vertices and the lines which are used to connect one node to the other nodes are called edges.
A DEGREE of a NODE is the number of edges connected to that node. A graph can be represented by G = (V,E), the presentation is a tuple which contains two sets. V is the set of vertices and E is the set of edges. However, E is decided based on the vertices which means the E is represented by the connection between the vertices. An example of a graph , G = ({1,2,3,4},{(1,2),(2,3),(3,4)}) here the first part presents the vertices so there are in total 4 vertices. And the later part is the vertices that are joined to one another. For (1,2) this means vertice 1 is connected with 2 and so on. As we can see the connection between vertices is shown as tuples so (1,2) is not equal to (2,1) and vice versa.
Therefore, for an undirected graph, we need to list all the connections for the same vertices. However, for directed graphs, writing it in the opposite way is not correct if that specific connection is not present in the graph. To disregard the rewriting for {(1,2), (2,1)}, we can mention that it is an Undirected Graph and instead write {(1,2)}.
Subgraphs are a portion of a graph. In other words, the vertices and edges are subsets of the graph.
PATH is the sequence of edges.
A graph is said to be CYCLE when the vertices are connected in such a way that it forms a cycle. We can go from one vertex to another vertex and return to the original starting vertex because all the vertices are connected in a closed cycle.
In a CONNECTED graph there must be a path from one vertex to all other vertices.Therefore, a graph is said to be connected if we choose a vertex and go to any other vertex.

A TREE is another form of graph but with a hierarchical structure. Here, there is a root node at the very top and from that node, more nodes are connected to other nodes in a top to bottom format. Thus , any two vertices are connected by exactly one path. Leaves are the nodes which do not have any other nodes connected with it so no edges are found after those nodes. A tree is said to be acyclic which means it does not form any cycle unlike graphs.
In DIRECTED graphs, there are directed edges. So the direction from one vertex to another is defined. Outdegree is the number of edges out of a specific node and In degree is the number of edges going inside a node. A directed path is a sequence of vertices joined to each other through directed edges in which all the edges are directed in the same path. A strong;y connected graph is when all vertices have paths to each other vertices.

Alphabets and strings, languages:

Alphabets are the building blocks of languages. In automata, alphabets are said to be a finite set of symbols. An example:

Σ_1 = {a, b, c, d, …, z}: the set of letters in English

Σ_2 = {0, 1, …, 9}: the set of (base 10) digits

Σ_3 = {a, b, …, z, #}: the set of letters plus the
special symbol #

Here, Σ sigma is used to formally represent the sets of alphabets.

From the examples we can see that the numbers are also alphabets and the symbols ?,! - and so on can also be called as alphabets if they are symbols.

On the other hand, a string is a finite sequence of symbols from the alphabet sigma. Which means a string is made up of symbols which are present in given Σ sigma only. for example, acd is a string over the alphabet Σ_1 = {a, b, c, d, …, z} because 'acd' each symbol is present inside the set. However, for ac$d is not a string over the alphabet Σ_1 = {a, b, c, d, …, z} because '$' in the string is not present in the set given. An empty string has no symbols present in the string and it is denoted by ε. Σ* This symbol represents all the strings over the given sigma Σ. For example, the set is given Σ_1 = {a, b, c, d, …, z} and the Σ1*= {a, b, c, d, …, z, ab, ac, ad,......} all the possible strings will be listed.

Languages:
Languages are a set of strings but certains rules/conditions are applied to it. Therefore, combinations of strings within some sets of restrictions such as grammar,expressions which produce a language. For example, $L_1 = \{x \in \Sigma_1^*$: x contains the substring "to"} , here L1 is the language and the restrictions or criteria that must be followed is that the string must contain the substring "to". x is the string taken from the sets of all strings Σ_1^*. If we recall, Σ_1 = {a,b,c,...z} and the Σ_1^* will contain all the possible combinations of symbols from sigma 1. So Σ_1^* ={a,b,c,...,z,ab,ac,...}. stop, to, toe are in L_1 and ε, oyster are not in L_1 because "to" is not present although in oyster "t" and "o" are there; however, they exactly need to be together "to"to form the required language.
Some more examples of languages:

1. L_2 = $\{x \in \Sigma_2^*$: x is divisible by 7}

   For this language 2 , it will have all the elements from all strings for which all the possible strings are made up of {1,2,3,..9}. So, Σ_2^* = {1,2,3,4,5,6,7,8,9,.....,24,25….} However the criteria that needs to be followed is that the elements(string) taken from that set must be divisible by 7 to form a language. Here is the final answer, {7, 14, 21, …}.

2. L_3 = $\{s\#s: s \in \{a, b, \ldots, z\}^*\}$

   For this language, the criteria is that s which is the string taken from the … , there must be "#" in between the two s and both the s on the left and right side must be equal.
   So for example, if we say we#ew is a language? The answer is no because as the criteria

mentioned above that s at both side of # needs to be exactly the same.

Some terms:
Suppose there is a string R = acbd so,

1. Length - |R| = 4
2. Empty string - here the sequence has length zero which means no symbols are present in the string.
3. Substring - substring of R can be a ,c , b, d, ac , cd, bd, acb, cbd, acbd however, if we write cbda we cannot say it is a substring of R because we know that
4. Prefix - in all the substrings the first symbol must be the same from the first symbol taken from the string. In other words, substrings that must contain the first symbol of a string
5. Suffice- substrings that must contain the last symbol of a string.
6. Ordering or sorting strings - sorting of symbols according to their value.
7. Concatenation- joining of two strings however, here the sequence does matter. For example, S = xyz so , R+S = acbdxyz and S+R = xyzacbd.
8. Reverse: acbd -> dbca
9. Equality: The sequence or position of each symbol is important here. acbd=acbd but not equal to abcd just because the position of the symbols b and c are interchanged.