**Data Structures**

Data Structures are the programmatic way of storing data so that data can be used efficiently. Following terms are the foundation terms of a data structure -

* **Interface** − Each data structure has an interface. Interface represents the set of operations that a data structure supports. An interface only provides the list of supported operations, type of parameters they can accept and return type of these operations.
* **Implementation** − Implementation provides the internal representation of a data structure. Implementation also provides the definition of the algorithms used in the operations of the data structure.

## **Characteristics of a Data Structure-**

* **Correctness** − Data structure implementation should implement its interface correctly.
* **Time Complexity** − Running time or the execution time of operations of data structure must be as small as possible.
* **Space Complexity** − Memory usage of a data structure operation should be as little as possible.

## **Need for Data Structure-**

As applications are getting complex and data rich, there are three common problems that applications face now-a-days-

* **Data Search** − Consider an inventory of 1 million(106) items of a store. If the application is to search an item, it has to search an item in 1 million(106) items every time slowing down the search. As data grows, search will become slower.
* **Processor speed** − Processor speed although being very high, falls limited if the data grows to billion records.
* **Multiple requests** − As thousands of users can search data simultaneously on a web server, even the fast server fails while searching the data.

## **Basic Terminology-**

* **Data** − Data are values or set of values.
* **Data Item** − Data item refers to single unit of values.
* **Group Items** − Data items that are divided into sub items are called as Group Items.
* **Elementary Items** − Data items that cannot be divided are called as Elementary Items.
* **Attribute and Entity** − An entity is that which contains certain attributes or properties, which may be assigned values.
* **Entity Set** − Entities of similar attributes form an entity set.
* **Field** − Field is a single elementary unit of information representing an attribute of an entity.
* **Record** − Record is a collection of field values of a given entity.
* **File** − File is a collection of records of the entities in a given entity set

**Algorithms-**

Algorithm is a step-by-step procedure, which defines a set of instructions to be executed in a certain order to get the desired output. Algorithms are generally created independent of underlying languages

From the data structure point of view, following are some important categories of algorithms −

* **Search** − Algorithm to search an item in a data structure.
* **Sort** − Algorithm to sort items in a certain order.
* **Insert** − Algorithm to insert item in a data structure.
* **Update** − Algorithm to update an existing item in a data structure.
* **Delete** − Algorithm to delete an existing item from a data structure.

## **Characteristics of an Algorithm-**

Not all procedures can be called an algorithm. An algorithm should have the following characteristics −

* **Unambiguous** − Algorithm should be clear and unambiguous. Each of its steps (or phases), and their inputs/outputs should be clear and must lead to only one meaning.
* **Input** − An algorithm should have 0 or more well-defined inputs.
* **Output** − An algorithm should have 1 or more well-defined outputs, and should match the desired output.
* **Finiteness** − Algorithms must terminate after a finite number of steps.
* **Feasibility** − Should be feasible with the available resources.
* **Independent** − An algorithm should have step-by-step directions, which should be independent of any programming code.

## **Algorithm Analysis-**

Efficiency of an algorithm can be analyzed at two different stages, before implementation and after implementation. They are the following −

* **A Priori Analysis** − This is a theoretical analysis of an algorithm. Efficiency of an algorithm is measured by assuming that all other factors, for example, processor speed, are constant and have no effect on the implementation.
* **A Posterior Analysis** − This is an empirical analysis of an algorithm. The selected algorithm is implemented using programming language. This is then executed on target computer machine. In this analysis, actual statistics like running time and space required, are collected.