

Data Structures and Algorithms

Why to Learn Data Structure and Algorithms?

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(10^6) items of a store. If the application is to search an item, it has to search an item in 1 million(10^6) 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.

To solve the above-mentioned problems, data structures come to rescue. Data can be organized in a data structure in such a way that all items may not be required to be searched, and the required data can be searched almost instantly.

What is Data Structure?

Data Structure is a way to store and organize data so that it can be used efficiently.

A data structure is defined as a particular way of storing and organizing data in our devices to use the data efficiently and effectively. The main idea behind using data structures is to minimize the time and space complexities. An efficient data structure takes minimum memory space and requires minimum time to execute the data.

The data structure name indicates itself that organizing the data in memory. There are many ways of organizing the data in the memory as we have already seen one of the data structures, i.e., array in C++ language. Array is a collection of memory elements in

which data is stored sequentially, i.e., one after another. In other words, we can say that array stores the elements in a continuous manner.

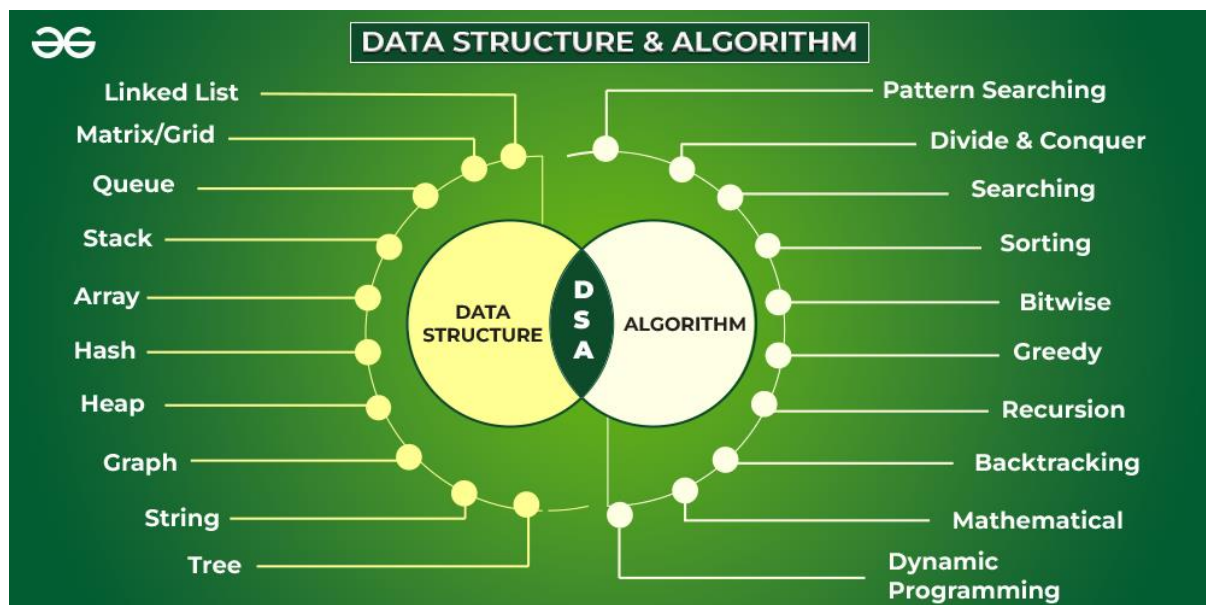
The data structure is not any programming language like C, C++, java, etc. It is a set of algorithms that we can use in any programming language to structure the data in the memory.

What is Algorithm?

Algorithm is defined as a process or set of well-defined instructions that are typically used to solve a particular group of problems or perform a specific type of calculation. To explain in simpler terms, it is a set of operations performed in a step-by-step manner to execute a task.

What is DSA?

The term DSA stands for **Data Structures and Algorithms**. As the name itself suggests, it is a combination of two separate yet interrelated topics – Data Structure and Algorithms.



Types of Data Structures

There are two types of data structures:

- Primitive data structure
- Non-primitive data structure

Primitive Data structure

The primitive data structures are primitive data types. The int, char, float, double, and pointer are the primitive data structures that can hold a single value.

Non-Primitive Data structure

The non-primitive data structure is divided into two types:

- Linear data structure
- Non-linear data structure

Linear Data Structure

The arrangement of data in a sequential manner is known as a linear data structure. The data structures used for this purpose are Arrays, Linked list, Stacks, and Queues. In these data structures, one element is connected to only one another element in a linear form.

Linear Data Structure

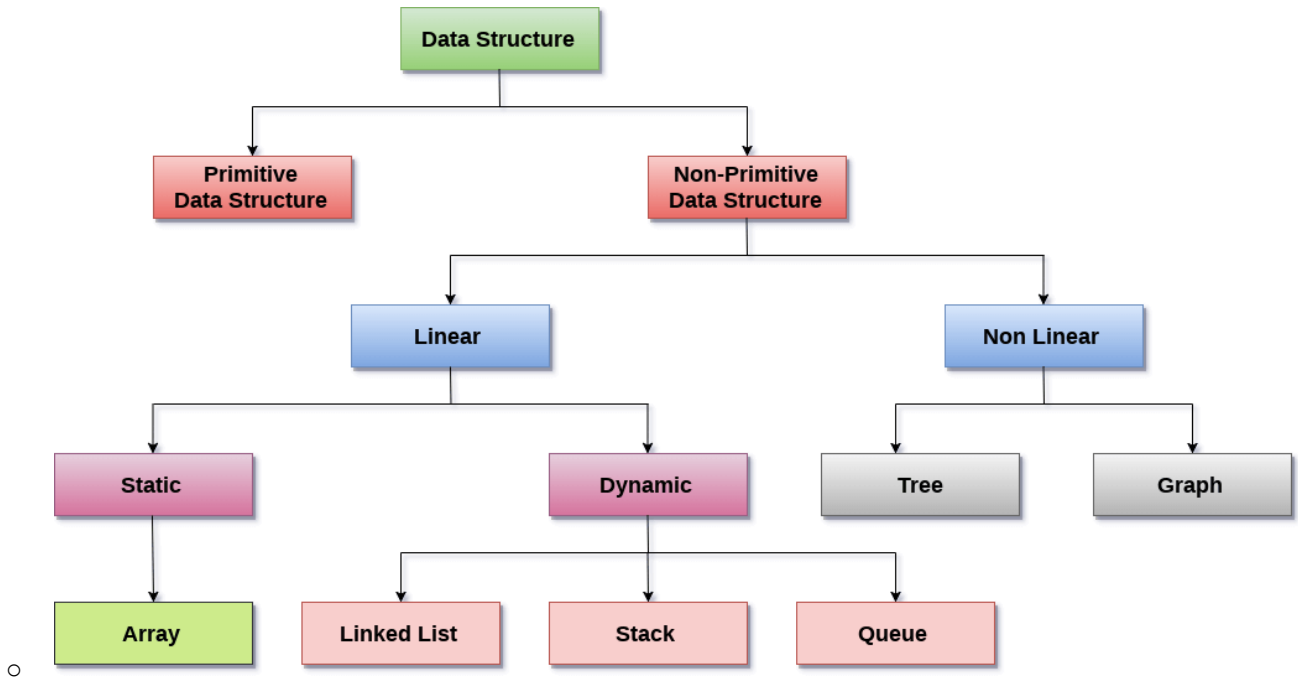
When one element is connected to the 'n' number of elements known as a non-linear data structure. The best example is trees and graphs. In this case, the elements are arranged in a random manner.

We will discuss the above data structures in brief in the coming topics. Now, we will see the common operations that we can perform on these data structures.

Data structures can also be classified as:

- **Static data structure:** It is a type of data structure where the size is allocated at the compile time. Therefore, the maximum size is fixed.
- **Dynamic data structure:** It is a type of data structure where the size is allocated at the run time. Therefore, the maximum size is flexible.

Types of Data Structures



Major Operations

The major or the common operations that can be performed on the data structures are:

- **Searching:** We can search for any element in a data structure.
- **Sorting:** We can sort the elements of a data structure either in an ascending or descending order.
- **Insertion:** We can also insert the new element in a data structure.
- **Updation:** We can also update the element, i.e., we can replace the element with another element.
- **Deletion:** We can also perform the delete operation to remove the element from the data structure.
- **Traversing:** Every data structure contains the set of data elements. Traversing the data structure means visiting each element of the data structure in order to perform some specific operation like searching or sorting.