Unit 2: Data Structures Notes – 1

From here onwards we enter the section of your course that deals with not only programming but also the most important trait of a developer, that is, logic and problem solving.

We would be introduced and limit ourselves to some basic problems for the sake of the course but as you go to higher classes you will encounter more complex problems and trying to solve them will make you a better developer.

Introduction to Data Structures:

Storing, Retrieving, Using and then again Storing data are the basis of nearly every application and program that we see in the internet. From your phone to the moon data plays a vital role in all the operations that are performed.

Now the data should also be stored efficiently so that accessing it becomes easy. Take an example of your cupboard. If you keep the cupboard disorganized you will have trouble finding let's, say a particular dress that you want to wear. But opposite to this, if you organize your cupboard into sections such as party wear, home wear, summer wear, winter wear etc. you know exactly in which section the dress is in, and though you have to search for it, it takes less time in comparison.

This is the primary use of data structures. They are used to store and retrieve information/data efficiently (not in all cases) and as fast as possible.

Definition: The study of data structure allows us to understand the organization of data and the management of the data flow in order to increase the efficiency of any process or program. Data Structure is a particular way of storing and organizing data in the memory of the computer so that these data can easily be retrieved and efficiently utilized in the future when required.

There are two primary classifications of Data Structures in Computer Science:

1. Primitive Data Structure: Primitive Data Structures directly operate according to the machine instructions. These are the primitive data types. Data types like int, char, float, double, and pointer are primitive data structures that can hold a single value.

2. Non – Primitive Data Structures: Non-primitive data structures are complex data structures that are derived from primitive data structures. These are the data structures that take away the sleep of developers and these are the ones we would be specially focus on. These are divided further into two categories:

a. Linear Data Structure:
   Best Example:

   Definition: Linear Data Structure consists of data elements arranged in a sequential manner where every element is connected to its previous and next elements. This connection helps to traverse a linear arrangement in a single level and in a single run.
   Some examples are: Array, Linked List, Queue, Stack etc.

   For the sake of the course we would get a basic introduction of different Linear Data Structures and how do they function.

b. Non-Linear Data Structure:
   Example:

   Definition: Non-linear Data Structures do not have any set sequence of connecting all its elements and every element can have multiple paths to attach to other elements. Such data structures support multi-level storage and sometimes can't be traversed in a single run. Some examples are Trees, Graphs, BSTs etc.

On Further Classification of the Data Structure, we get:

a. Static Data Structure: In these kinds of data structures the size of the data structure is allocated during compile time and the maximum size of the structure cannot be changed.
b. Dynamic Data Structure: Here the size is allocated at run time and hence the maximum size is flexible and can be changes as per requirement.