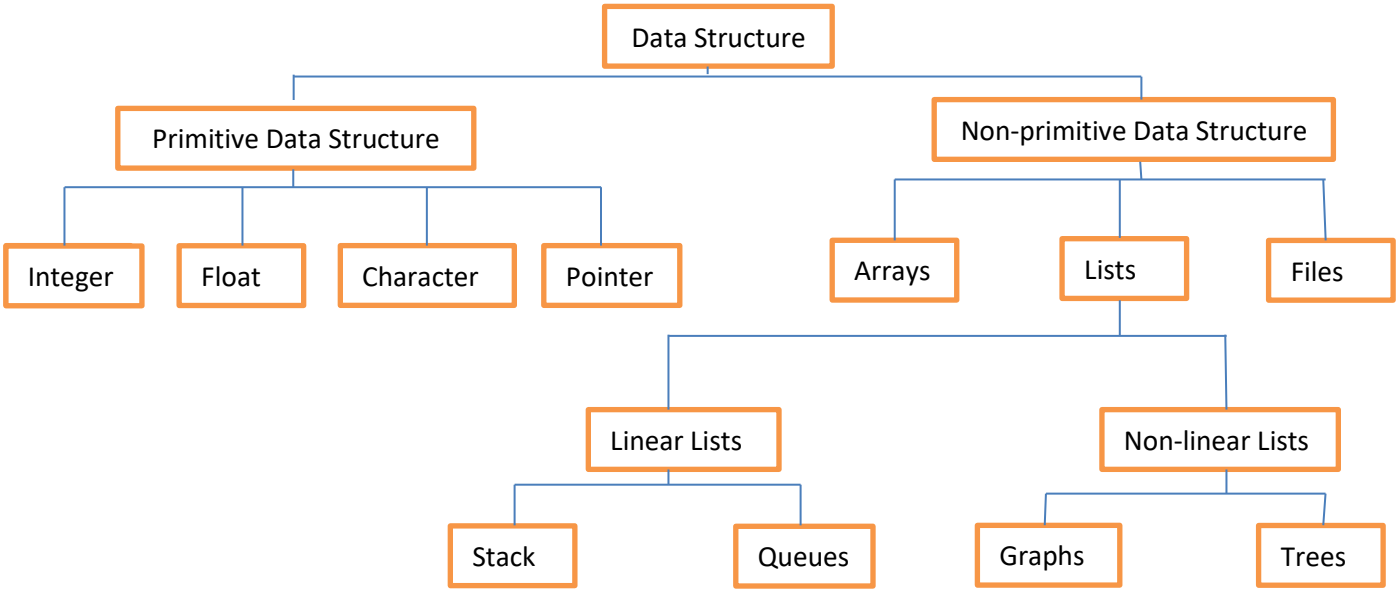
INTRODUCTION

- **Data Structure**: - A data structure is a way of organizing all data items that considers not only the elements stored but also their relationship to each other.
  In other words, we can say that the logical or mathematical model of a particular organization of data items is called data structure.
- **Data Structure mainly specifies (or dictates) the following four things**:
  I. Organization of data
  II. Accessing methods
  III. Degree of associativity
  IV. Processing alternatives for information
- **Data Structures are the building blocks of a program and hence the selection of a particular data structure stresses on the following two things**:
  1. The data structures must be rich enough in structure to reflect the relationship existing between the data.
  2. And the structure should be simple so that we can process data effectively whenever required.
- The structure of input and output data can be used to derive the structure of a program. Data structure affects the design of both structural and functional aspects of a program.

  Algorithm + Data Structure = Program

  To develop a program of an algorithm, we should select an appropriate data structure for that algorithm. Therefore, algorithm and its associated data structures form a program.
- **Classification of Data Structure**:
  Data Structures are normally divided into two broad categories:
  I. Primitive data structures
  II. Non-primitive data structures.
- **Primitive Data Structure**: - These are basic structures and are directly operated upon by the machine instructions. These, in general have different representations on different computers. Integer, floating-point numbers, character constants, string constants, pointers etc. fall in this category.



- **Non-Primitive Data Structures**: - These are derived from the primitive data structures. The non-primitive data structures emphasize on structuring of a group of homogeneous (same type) or heterogeneous (different type) data items. Arrays, lists and files are examples.
  The most commonly used operations on data structures are broadly categorized into four types.
  a. **CREATE**: - The **CREATE** operation results in reserving memory for the program elements. This can be done by **declaration statement**. The creation of data structure may take place either during **compile-time** or during **run-time**.
  b. **DESTROY or DELETE**: - **DESTROY** operation destroys the memory space allocated for the specified data structure.
  malloc() and **free()** function of **C language** are used for **CREATE** and **DESTROY** operations respectively (as far as dynamic memory allocation and deallocation is concerned).
  c. **SELECTION**: - The **SELECTION** operation deals with accessing a particular **data** within a data structure.
  d. **UPDATION**: - It updates or modifies the data in the data structure. Probably new data may be entered or previously stored data may be deleted.
  Other operations performed on data structures include:
  i. **Searching**: - Searching operation finds the presence of the desired data item in the list of data item. It may also find the locations of all elements that satisfy certain conditions.

ii. **Sorting**: - Sorting is the process of arranging all data items in a data structure in a particular order say for example, either in ascending order or in descending order.

iii. **Merging**: - Merging is a process of combining the data items of two different sorted lists into a single sorted list.

- **Arrays**: - An array is defined as a set of finite number of homogeneous elements or data items that stored in contiguous memory location. It means an array can contain one type of data only, either all integers, all floating-point numbers, or all characters. Declaration of arrays is as follows :

  int arr[10];

  Where **int** specifies the data type or type of elements array stores. **"arr"** is the name of array, and the number specified inside the square brackets is the number of elements an array can store, this is also called size or length of array.