

# DATA STRUCTURES



Sharea **Coke**  
Sharea **feeling**



FIND OUT HOW

[SEE THE INDEX](#)



## Introduction to Data Structures

Data Structure is a way of collecting and organising data in such a way that we can perform operations on these data in an effective way. Data Structures is about rendering data elements in terms of some relationship, for better organization and storage. For example, we have data player's name "Virat" and age 26. Here "Virat" is of **String** data type and 26 is of **integer** data type.

We can organize this data as a record like **Player** record. Now we can collect and store player's records in a file or database as a data structure. For example: "Dhoni" 30, "Gambhir" 31, "Sehwag" 33

In simple language, Data Structures are structures programmed to store ordered data, so that various operations can be performed on it easily.

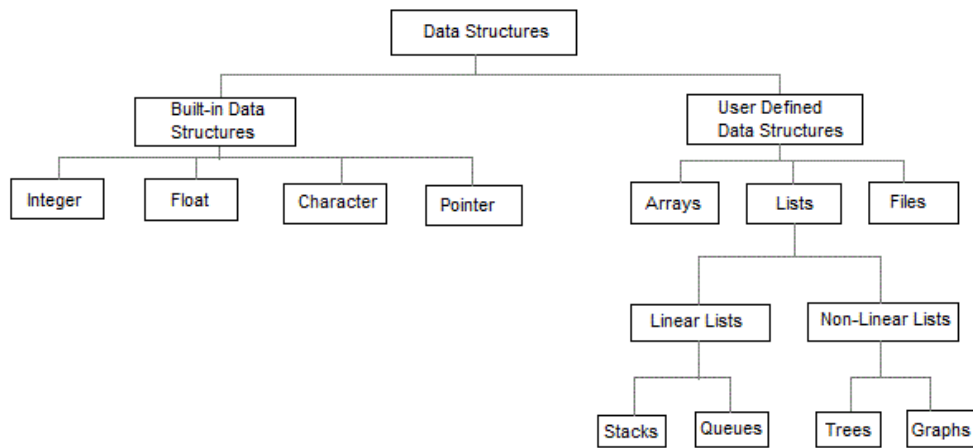
### Basic types of Data Structures

As we discussed above, anything that can store data can be called as a data structure, hence Integer, Float, Boolean, Char etc, all are data structures. They are known as **Primitive Data Structures**.

Then we also have some complex Data Structures, which are used to store large and connected data. Some example of **Abstract Data Structure** are :

- Linked List
- Tree
- Graph
- Stack, Queue etc.

All these data structures allow us to perform different operations on data. We select these data structures based on which type of operation is required. We will look into these data structures in more details in our later lessons.



## INTRODUCTION TO DATA STRUCTURES

### What is Algorithm ?

An algorithm is a finite set of instructions or logic, written in order, to accomplish a certain predefined task. Algorithm is not the complete code or program, it is just the core logic(solution) of a problem, which can be expressed either as an informal high level description as **pseudocode** or using a **flowchart**.

An algorithm is said to be efficient and fast, if it takes less time to execute and consumes less memory space. The performance of an algorithm is measured on the basis of following properties :

1. Time Complexity
2. Space Complexity

### Space Complexity

Its the amount of memory space required by the algorithm, during the course of its execution. Space complexity must be taken seriously for multi-user systems and in situations where limited memory is available.

An algorithm generally requires space for following components :

- **Instruction Space** : Its the space required to store the executable version of the program. This space is fixed, but varies depending upon the number of lines of code in the program.
- **Data Space** : Its the space required to store all the constants and variables value.
- **Environment Space** : Its the space required to store the environment information needed to resume the suspended function.

### Time Complexity

Time Complexity is a way to represent the amount of time needed by the program to run to completion. We will study this in details in our section.

**NOTE:** Before going deep into data structure, you should have a good knowledge of programming either in C or in C++ or Java.

Next → (time-complexity-of-algorithms)

