

[Home](#)   [Data Structure](#)   [C](#)   [C++](#)   [C#](#)   [Java](#)   [SQL](#)   [HTML](#)   [CSS](#)   [JavaScript](#)   [Ajax](#)   [Android](#)   [Cloud](#)   [Design Pattern](#)   [Quiz](#)

ADVERTISEMENT BY ADRECOVER

↑ SCROLL TO TOP

# Data Structure

## Introduction

Data Structure can be defined as the group of data elements which provides an efficient way of storing and organising data in the computer so that it can be used efficiently. Some examples of Data Structures are arrays, Linked List, Stack, Queue, etc. Data Structures are widely used in almost every aspect of Computer Science i.e. Operating System, Compiler Design, Artificial intelligence, Graphics and many more.

Data Structures are the main part of many computer science algorithms as they enable the programmers to handle the data in an efficient way. It plays a vital role in enhancing the performance of a software or a program as the main function of the software is to store and retrieve the user's data as fast as possible.

## Basic Terminology

Data structures are the building blocks of any program or the software. Choosing the appropriate data structure for a program is the most difficult task for a programmer. Following terminology is used as far as data structures are concerned.

**Data:** Data can be defined as an elementary value or the collection of values, for example, student's name and its id are the data about the student.

**Group Items:** Data items which have subordinate data items are called Group item, for example, name of a student can have first name and the last name.

↑ SCROLL TO TOP

**Record:** Record can be defined as the collection of various data items, for example, if we talk about the student entity, then its name, address, course and marks can be grouped together to form the record for the student.

**File:** A File is a collection of various records of one type of entity, for example, if there are 60 employees in the class, then there will be 20 records in the related file where each record contains the data about each employee.

**Attribute and Entity:** An entity represents the class of certain objects. it contains various attributes. Each attribute represents the particular property of that entity.

**Field:** Field is a single elementary unit of information representing the attribute of an entity.

## Need of Data Structures

As applications are getting complexed and amount of data is increasing day by day, there may arise the following problems:

**Processor speed:** To handle very large amount of data, high speed processing is required, but as the data is growing day by day to the billions of files per entity, processor may fail to deal with that much amount of data.

**Data Search:** Consider an inventory size of 106 items in a store, If our application needs to search for a particular item, it needs to traverse 106 items every time, results in slowing down the search process.

**Multiple requests:** If thousands of users are searching the data simultaneously on a web server, then there are the chances that a very large server can be failed during that process

in order to solve the above problems, data structures are used. Data is organized to form a data structure in such a way that all items are not required to be searched and required data can be searched instantly.

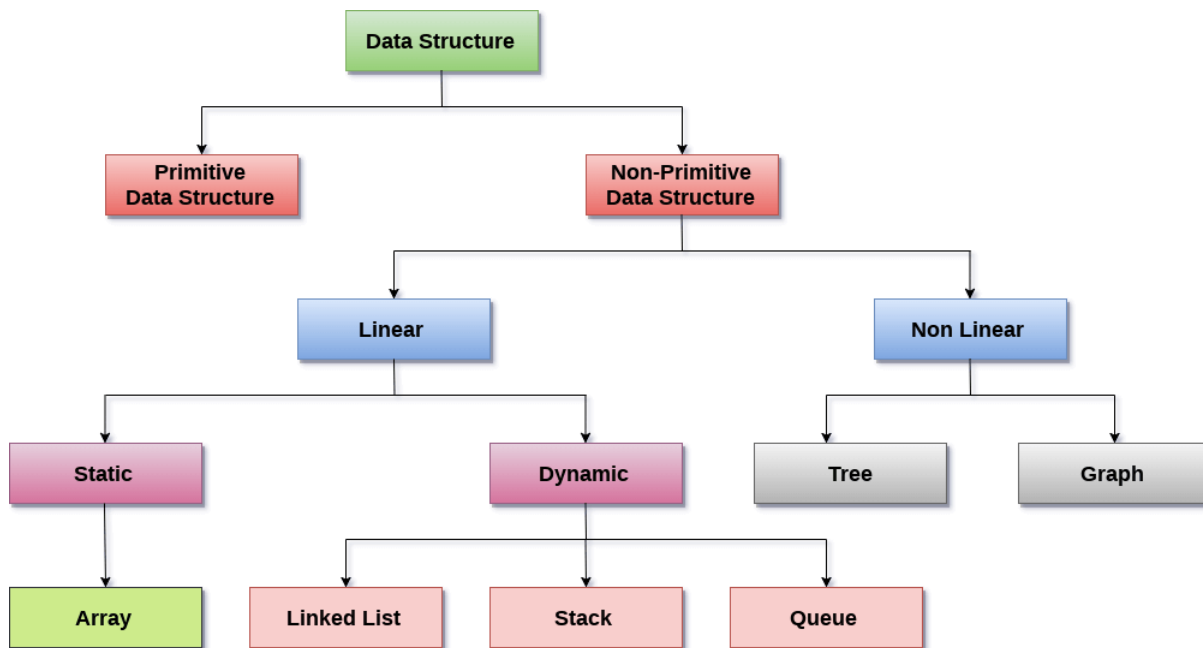
## Advantages of Data Structures

**Efficiency:** Efficiency of a program depends upon the choice of data structures. For example: suppose, we have some data and we need to perform the search for a particular record. In that case, if we organize our data in an array, we will have to search sequentially element by element. hence, using array may not be very efficient here. There are better data structures which can make the search process efficient like ordered array, binary search tree or hash tables.

**Reusability:** Data structures are reusable, i.e. once we have implemented a particular data structure, we can use it at any other place. Implementation of data structures can be compiled into libraries which can be used by different clients.

**Abstraction:** Data structure is specified by the ADT which provides a level of abstraction. The client program uses the data structure through interface only, without getting into the implementation details.

## Data Structure Classification



**Linear Data Structures:** A data structure is called linear if all of its elements are arranged in the linear order. In linear data structures, the elements are stored in non-hierarchical way where each element has the successors and predecessors except the first and last element.

Types of Linear Data Structures are given below:

**Arrays:** An array is a collection of similar type of data items and each data item is called an element of the array. The data type of the element may be any valid data type like char, int, float or double.

The elements of array share the same variable name but each one carries a different index number known as subscript. The array can be one dimensional, two

↑ SCROLL TO TOP    dimensional.

The individual elements of the array age are:

age[0], age[1], age[2], age[3],..... age[98], age[99].

**Linked List:** Linked list is a linear data structure which is used to maintain a list in the memory. It can be seen as the collection of nodes stored at non-contiguous memory locations. Each node of the list contains a pointer to its adjacent node.

**Stack:** Stack is a linear list in which insertion and deletions are allowed only at one end, called **top**.

A stack is an abstract data type (ADT), can be implemented in most of the programming languages. It is named as stack because it behaves like a real-world stack, for example: - piles of plates or deck of cards etc.

**Queue:** Queue is a linear list in which elements can be inserted only at one end called **rear** and deleted only at the other end called **front**.

It is an abstract data structure, similar to stack. Queue is opened at both end therefore it follows First-In-First-Out (FIFO) methodology for storing the data items.

**Non Linear Data Structures:** This data structure does not form a sequence i.e. each item or element is connected with two or more other items in a non-linear arrangement. The data elements are not arranged in sequential structure.

Types of Non Linear Data Structures are given below:

**Trees:** Trees are multilevel data structures with a hierarchical relationship among its nodes. The bottommost nodes in the hierarchy are called **leaf** node. The topmost node is called **root node**. Each node contains pointers to

point adjacent nodes.

Tree data structure is based on the parent-child relationship among the nodes. Each node in the tree can have more than one children except the leaf nodes whereas each node can have at most one parent except the root node. Trees can be classified into many categories which will be discussed later in this tutorial.

**Graphs:** Graphs can be defined as the pictorial representation of the set of elements (represented by vertices) connected by the links known as edges. A graph is different from tree in the sense that a graph can have cycle while the tree can not have the one.

## Operations on data structure

1) **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.

**Example:** If we need to calculate the average of the marks obtained by a student in 6 different subjects, we need to traverse the complete array of marks and calculate the total sum, then we will divide that sum by the number of subjects i.e. 6, in order to find the average.

2) **Insertion:** Insertion can be defined as the process of adding the elements to the data structure at any location.

If the size of data structure is **n** then we can only insert **n-1** data elements into it.

3) **Deletion:** The process of removing an element from the data structure is called Deletion. We can delete an element from the data structure at any random location.

If we try to delete an element from an empty data structure then **underflow** occurs.

4) **Searching:** The process of finding the location of an element within the data structure is called Searching. There are two algorithms to perform searching, Linear Search and Binary Search. We will discuss each one of them later in this tutorial.

5) **Sorting:** The process of arranging the data structure in a specific order is known as Sorting. There are many algorithms that can be used to perform sorting, for example, insertion sort, selection sort, bubble sort, etc.

6) **Merging:** When two lists List A and List B of size M and N respectively, of similar type of elements, clubbed or joined to produce the third list, List C of size (M+N), then this process is called merging

← Prev

Next →

ADVERTISEMENT BY ADRECOVER

↑ SCROLL TO TOP





↓ow

## Feedback

- Send your Feedback to [feedback@javatpoint.com](mailto:feedback@javatpoint.com)

## Help Others, Please Share



## Learn Latest Tutorials

Digital Marketing

Elasticsearch



Entity Framework

↑ SCROLL TO TOP



Firewall



Functional  
Programming  
tutorial

Functional  
Programming



Google Colab  
tutorial

Google Colab



Graph Theory  
tutorial

Graph Theory



Groovy tutorial

Groovy



Group  
Discussion  
tutorial

Group Discussion



Informatica  
tutorial

Informatica



Ionic tutorial

Ionic



ITIL tutorial

ITIL



IOS  
Development  
tutorial

IOS with Swift



angular  
material tutorial

Angular Material



deep learning  
tutorial

Deep Learning

## Preparation



Antitude



Logical  
Reasoning  
Reasoning



Verbal Ability  
Verbal Ability

↑ SCROLL TO TOP



Interview  
Questions

Interview Questions



Company  
Interview  
Questions

Company Questions

## Trending Technologies



Artificial  
Intelligence  
Tutorial

Artificial  
Intelligence



AWS Tutorial  
AWS



Selenium  
tutorial

Selenium



Cloud  
Computing  
tutorial

Cloud Computing



Hadoop tutorial  
Hadoop



ReactJS  
Tutorial

ReactJS



Data Science  
Tutorial

Data Science



Angular 7  
Tutorial

Angular 7




Blockchain  
Tutorial

Blockchain

↑ SCROLL TO TOP


 Git Tutorial  
Git


 Machine  
Learning Tutorial  
Machine Learning


 DevOps  
Tutorial  
DevOps


## B.Tech / MCA

 DBMS tutorial  
DBMS


 Data Structures  
tutorial  
Data Structures


 DAA tutorial  
DAA

 Operating  
System tutorial  
Operating System

 Computer  
Network tutorial  
Computer Network

 Compiler  
Design tutorial  
Compiler Design


 Computer  
Organization and  
Architecture  
Computer  
Organization

 Discrete  
Mathematics  
Tutorial  
Discrete  
Mathematics

 Ethical Hacking  
Tutorial  
Ethical Hacking

 Computer

 Software  
Engineering  
Tutorial

 html tutorial  
Web Technology

↑ SCROLL TO TOP

Software  
Engineering

 Cyber Security  
tutorial


Cyber Security

 Automata  
Tutorial

Automata

 C Language  
tutorial

C Programming

 C++ tutorial

C++

 Java tutorial

Java

 .Net  
Framework  
tutorial


.Net

 Python tutorial

Python

 List of  
Programs

Programs

 Control  
Systems tutorial

Control System

 Data Mining  
Tutorial

Data Mining

 Data  
Warehouse  
Tutorial

Data Warehouse

ADVERTISEMENT BY ADRECOVER

↑ SCROLL TO TOP

