What is Data Structure: Types, Classifications

A data structure is a way to store data.

We structure data in different ways depending on what data we have, and what we want to do with it.

First, let's consider an example without computers in mind, just to get the idea.

If we want to store data about people we are related to, we use a family tree as the data structure. We choose a family tree as the data structure because we have information about people we are related to and how they are related, and we want an overview so that we can easily find a specific family member, several generations back.

With such a family tree data structure visually in front of you, it is easy to see, for example, who my mother's mother is—it is 'Emma,' right? But without the links from child to parents that this data structure provides, it would be difficult to determine how the individuals are related.

Data structures give us the possibility to manage large amounts of data efficiently for uses such as large databases and internet indexing services.

Data structures are essential ingredients in creating fast and powerful algorithms. They help in managing and organizing data, reduce complexity, and increase efficiency.

## **Characteristics of Data Structures**

Data Structure is the systematic way used to organise the data. The characteristics of Data Structures are:

### **Linear or Non-Linear**

This characteristic arranges the data in sequential order, such as arrays, graphs etc.

### **Static and Dynamic**

Static data structures have fixed formats and sizes along with memory locations. The static characteristic shows the compilation of the data.

### **Time Complexity**

The time factor should be very punctual. The running time or the execution time of a program should be limited. The running time should be as less as possible. The less the running time, the more accurate the device is.

### **Space Complexity**

The Space in the device should be managed carefully. The memory usage should be used properly. The space should be less occupied, which indicates the proper function of the device.

**Linear Data Structures**

Data elements in a linear data structure are linked to one another in a sequential arrangement, with each element linked to the elements in front of and behind it. In this manner, a single run can traverse the structure. Linear data structures consist of four types. They are:

* Stack
* Array
* Queue
* Linked list

### **Stack**

The linear data structure stores the data elements in the ‘first-in/ last-out’ or the ‘last-in/ first out’ order. These orders are known as FILO and LIFO orders, respectively. By using Stack, the element can be added and removed simultaneously from the same end. In Python, Stack can be developed in the following ways.

1. Queue.LifoQueue
2. List
3. Collections.deque

 In Stack, the terms ‘Push’ and ‘Pop’ are used instead of ‘insert’ and ‘delete’.

### **Array**

It is the collection of similar data types that are stored in the Contiguous Memory Locations. [Arrays](https://www.simplilearn.com/tutorials/data-structure-tutorial/arrays-in-data-structure) are used in Python as well. Arrays work on the scale of 0 to (n-1), where ‘n’ denotes the size of the array. Arrays are of two types. They are:

1. One-dimensional Array
2. Multi-dimensional Array

### **Queue**

The [queue](https://www.simplilearn.com/tutorials/data-structure-tutorial/queue-in-data-structure) is a linear data structure that follows the FIFO order. FIFO stands for First In and First Out. The order is that the elements which are inserted first are to be removed first. The properties of Queue data structure are:

1. Inserting an element
2. Deleting the element
3. Time of access.

### **Linked List**

[Linked Lists](https://www.simplilearn.com/tutorials/data-structure-tutorial/types-of-linked-list) separate the data structures that are stored consecutively. The last node of a data structure will be linked to the first node of the next data structure. The first element of any data structure is known as the Head of the List. The linked list helps in memory allocation, stores data in internal structure etc. There are three types of Linked Lists. They are:

1. Single Linked List
2. Double Linked List
3. Circular Linked List

## **Non-Linear Data Structures**

The data structure in which the data elements are randomly arranged. The elements are non-arranged sequentially. The data elements are present at different levels. In Non-linear data structures, there are different paths for an element to reach the other element. The data elements in the non-linear data structures are connected to one or more elements. There are two types of non-linear data structures. They are:

* Tree Data Structure
* Graph Data Structure

### **Tree Data Structure**

[Tree data structures](https://www.simplilearn.com/tutorials/data-structure-tutorial/trees-in-data-structure) are completely different from the arrays, stacks, queues and linked lists. Tree data structures are hierarchic. The tree data structure collects the nodes together to depict and stimulate the sequence. Tree data structure does not store the data sequentially. It stores the data on multiple levels. The top node of the Tree Data Structure is known as the Root Node. Any type of data can be stored in the root node. Each node shall definitely contain the data. The branches in the Tree Data Structure are known as the children.

The different parts of the Tree Data Structure are:

1. Root Node
2. Child Node
3. Edge
4. Siblings
5. Leaf Node
6. Internal Nodes
7. Height of the tree
8. Degree of the Node

### **Graph Data Structure**

In [Graph Data Structure](https://www.simplilearn.com/tutorials/data-structure-tutorial/graphs-in-data-structure), one node is simply connected to the other node through the edge of the graph. The Graph Data Structure obviously uses Non-linear data structures which are not sequentially arranged. The graph data structures consist of edges and nodes represented by E and V, respectively. Graph Data Structures do not have root nodes. It does not have a standard order of arranging the data. Every tree is also known as the graph with n-1 edges where ‘n’ represents the total number of vertices in the graph. There are various categories in the graphs such as undirected, unweighted, directed and weighted.

The different parts of the graph are as follows.

1. Vertex
2. Edges
3. Directed Edge
4. Undirected Edge
5. Weighted Edge
6. Degree
7. Indegree
8. Outdegree.

## **Linear Vs Non-linear Data Structures**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Linear Data Structure** | **Non-linear Data Structure** |
| Arrangement of the data elements | In the case of a linear data structure, the data items are stored in a linear order. Every element is linked to the first and next elements in the sequence. | In the case of a non-linear data structure, the data pieces are ordered non-linearly and attached hierarchically. The data elements are linked to several items. |
| Categories | A linear data structure can be an array, a stack, a linked list, or a queue. | Non-linear data structures include trees and graphs. |
| Levels | The linear data structure consists of a single level. It has no hierarchy. | There are several layers involved in this arrangement. As a result, the elements are organized hierarchically. |
| Traversal | Because linear data has only one level, traversing each data item needs only one run. | non-linear data structure data elements cannot be retrieved in a single run. It is necessary to traverse many runs. |
| Memory usages | Memory use is inefficient in this case. | Memory is used very efficiently in this case. |
| Applications | Linear data structures are mostly utilized in software development. | Image processing and artificial intelligence both make use of non-linear data structures. |
| Time Complexity | The time complexity of a linear data structure grows as the input size grows. | The time complexity of a non-linear data structure frequently remains constant as the input size increases. |
| Relationships | Only one form of relationship between the data pieces is possible. | A non-linear data structure can have a one-to-one or one-to-many connection between its pieces. |

## **Data Structure Operations**

The following are the most frequent operations that may be done on data structures:

1. Searching - Searching entails locating a certain piece inside a specified data structure. When the needed ingredient is discovered, it is termed a success. Searching is an operation that may be done on data structures such as arrays, linked lists, trees, graphs, and so on.
2. Sorting - Sorting is the process of ordering all data elements in a data structure in a certain order, such as ascending or descending order.
3. Insertion entails adding new data items to the data structure.
4. The data elements in the data structure can be deleted.
5. Updating - We can update or replace existing data structure parts

## **Characteristics of a Data Structure**

* Correctness -  Data structure implementation should accurately implement its interface.
* Time Complexity - The running time or execution time of data structure operations must be as short as feasible.
* Space Complexity - A data structure operation should use as little memory as feasible.