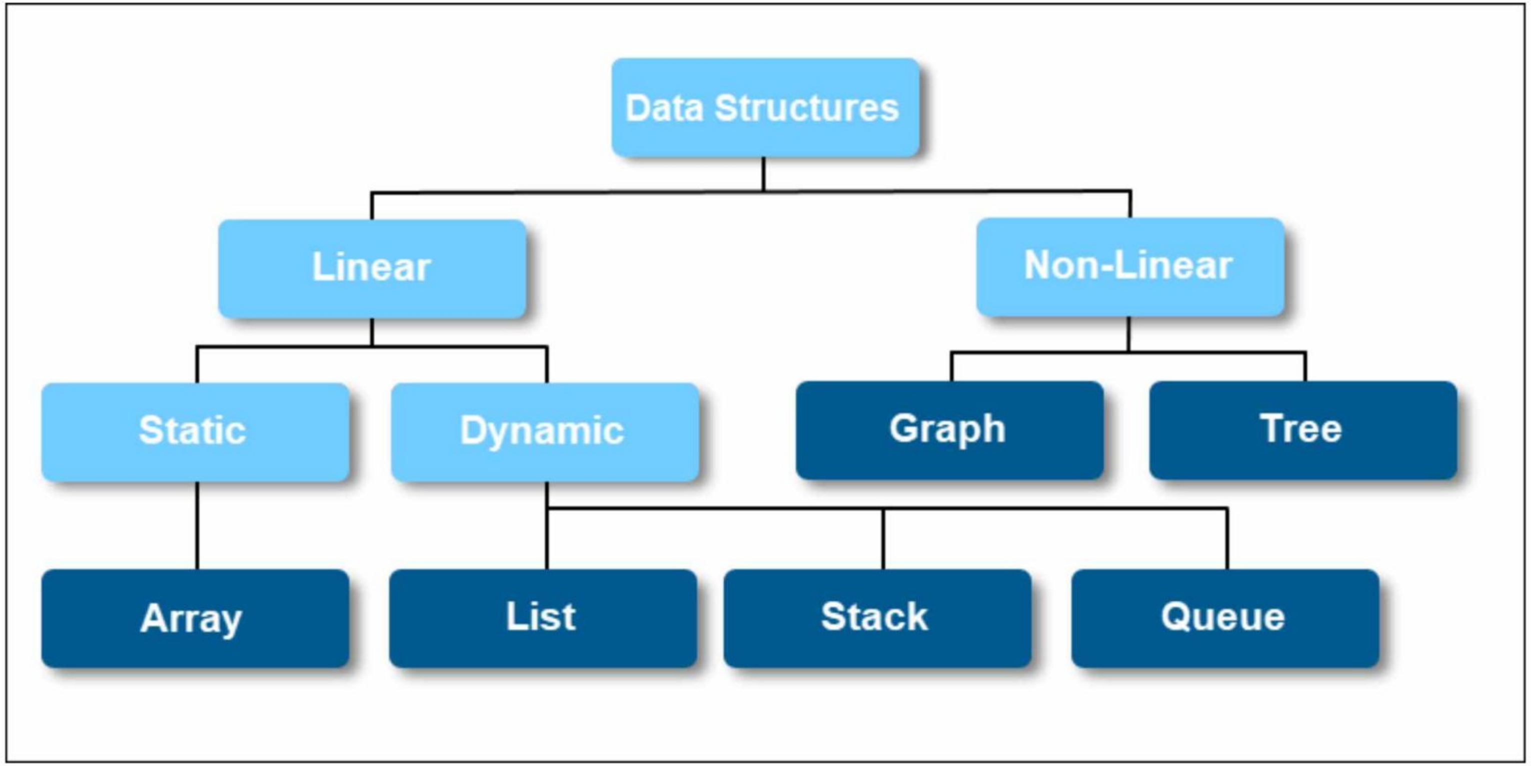
Introduction to Data Structures

Prepared By,
Saima Siddique Tashfia
Lecturer, Department of CSE
University of Information Technology & Sciences

What is Data Structure?

A data structure is a way of organizing and storing data in a computer so that it can be accessed and used efficiently. It refers to the logical or mathematical representation of data, as well as the implementation in a computer program

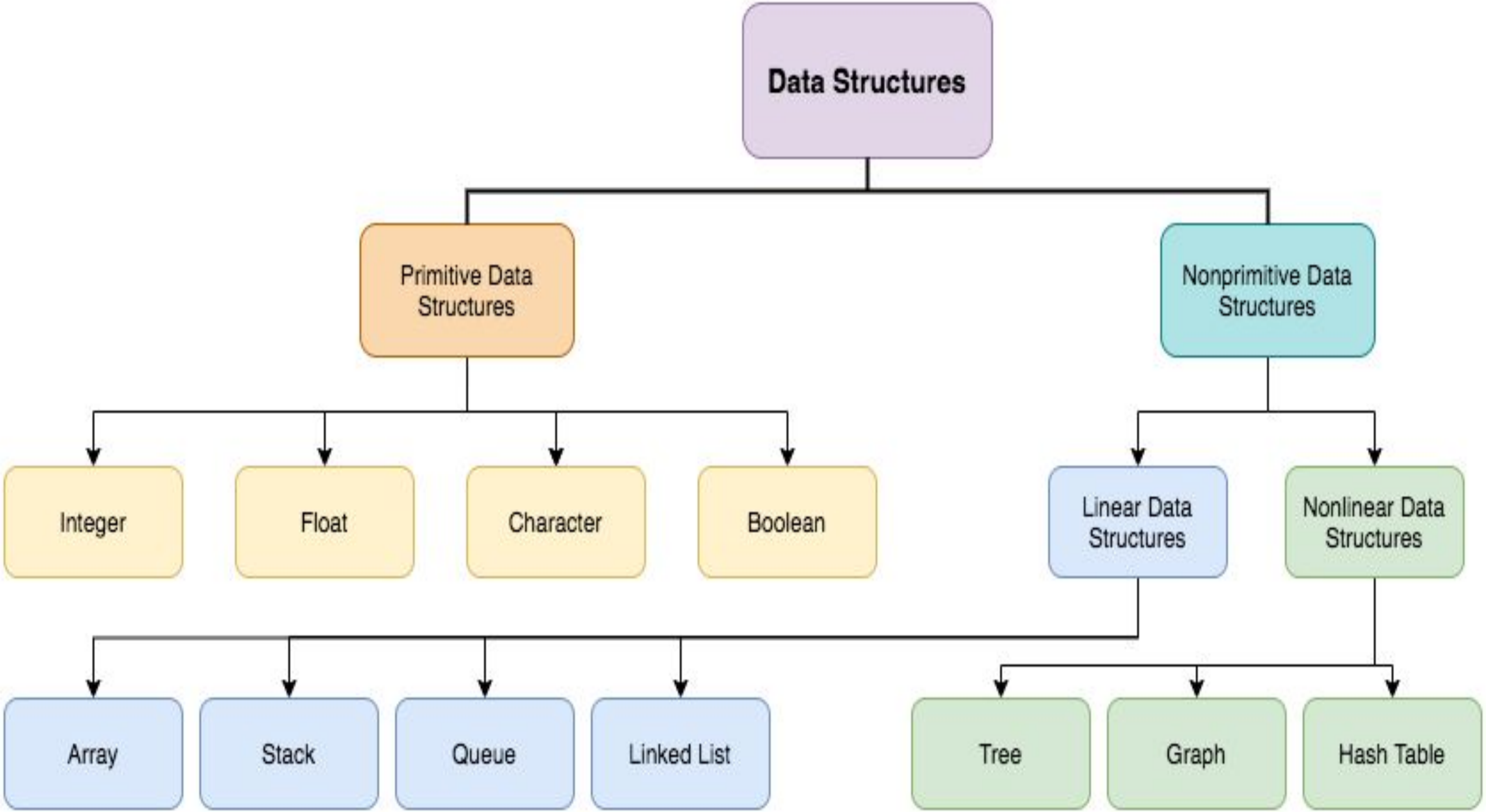
What is data structure?

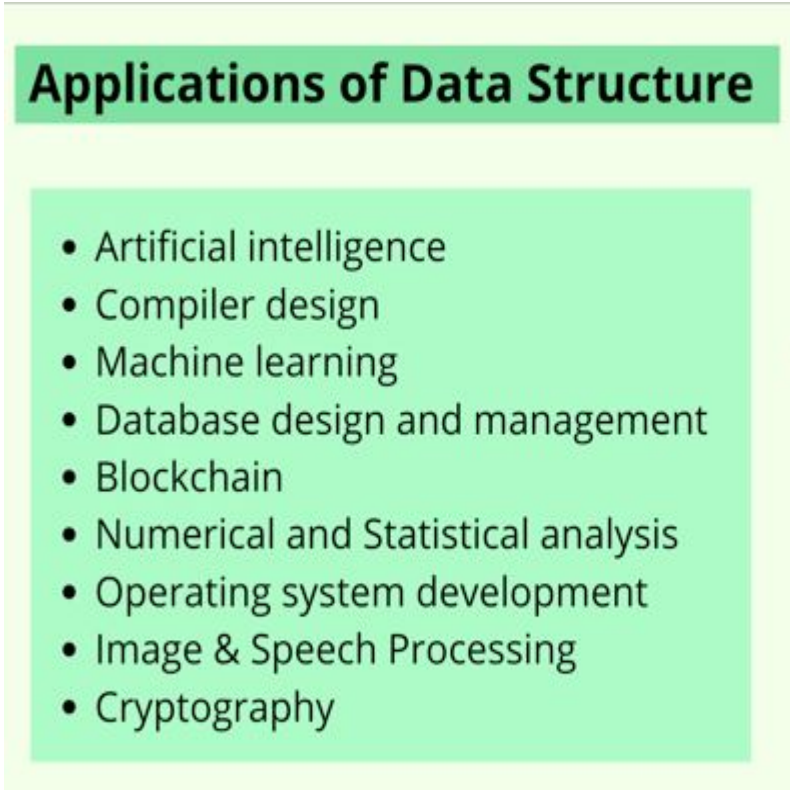


Classification

Data structures can be classified into two broad categories:

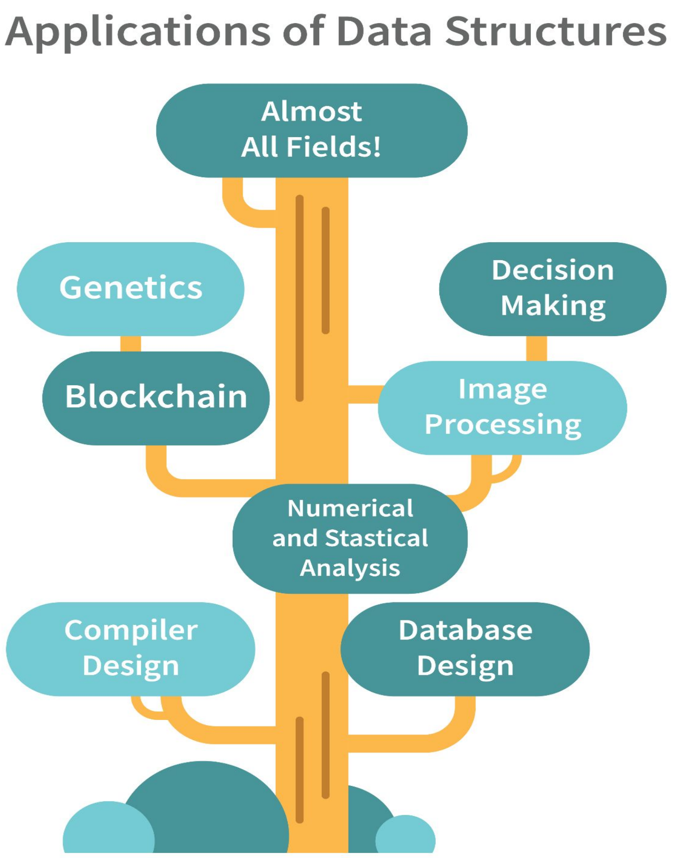
- **Linear Data Structure:** A data structure in which data elements are arranged sequentially or linearly, where each element is attached to its previous and next adjacent elements, is called a linear data structure. Examples are array, stack, queue, etc.
- **Non-linear Data Structure:** Data structures where data elements are not placed sequentially or linearly are called non-linear data structures. Examples are trees and graphs.


Data Structures
Primitive Data Structures
Nonprimitive Data Structures
Integer
Float
Character
Boolean
Linear Data Structures
Nonlinear Data Structures
Array
Stack
Queue
Linked List
Tree
Graph
Hash Table



Applications of Data Structure

- Artificial intelligence
- Compiler design
- Machine learning
- Database design and management
- Blockchain
- Numerical and Statistical analysis
- Operating system development
- Image & Speech Processing
- Cryptography

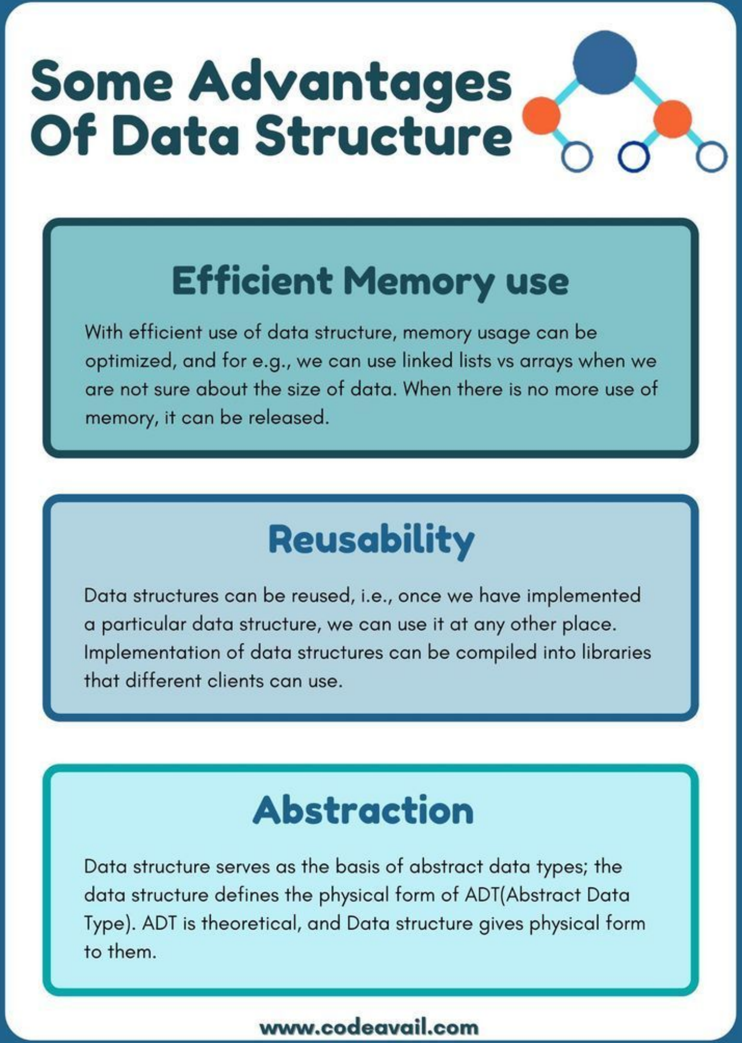


Data Structures

Advantages of Data Structures

The use of data structures provides several advantages, including:

- **Efficiency:** Data structures allow for efficient storage and retrieval of data, which is important in applications where performance is critical.
- **Flexibility:** Data structures provide a flexible way to organize and store data, allowing for easy modification and manipulation.
- **Reusability:** Data structures can be used in multiple programs and applications, reducing the need for redundant code.
- **Maintainability:** Well-designed data structures can make programs easier to understand, modify, and maintain over time.



Some Advantages Of Data Structure

Efficient Memory use

With efficient use of data structure, memory usage can be optimized, and for e.g., we can use linked lists vs arrays when we are not sure about the size of data. When there is no more use of memory, it can be released.

Reusability

Data structures can be reused, 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 that different clients can use.

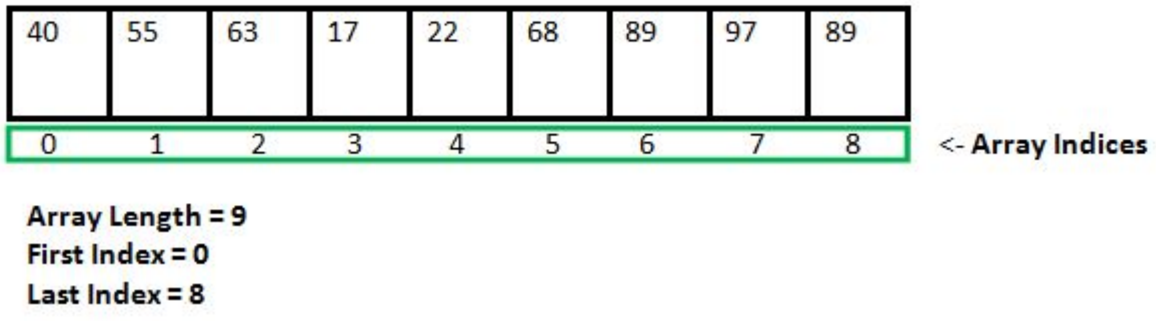
Abstraction

Data structure serves as the basis of abstract data types; the data structure defines the physical form of ADT(Abstract Data Type). ADT is theoretical, and Data structure gives physical form to them.

www.codeavail.com

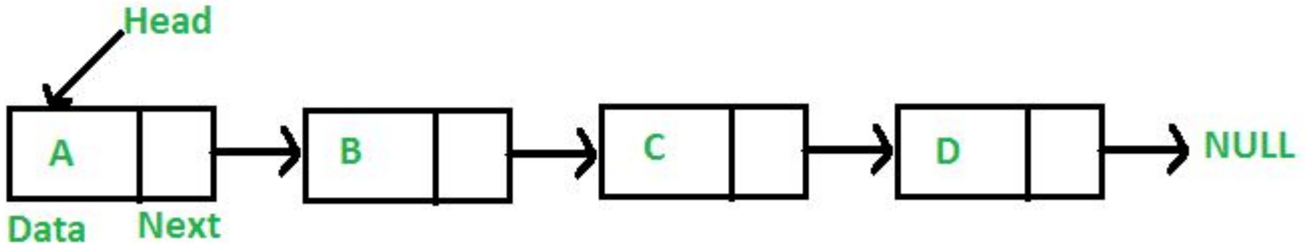
What is a Static Data structure?

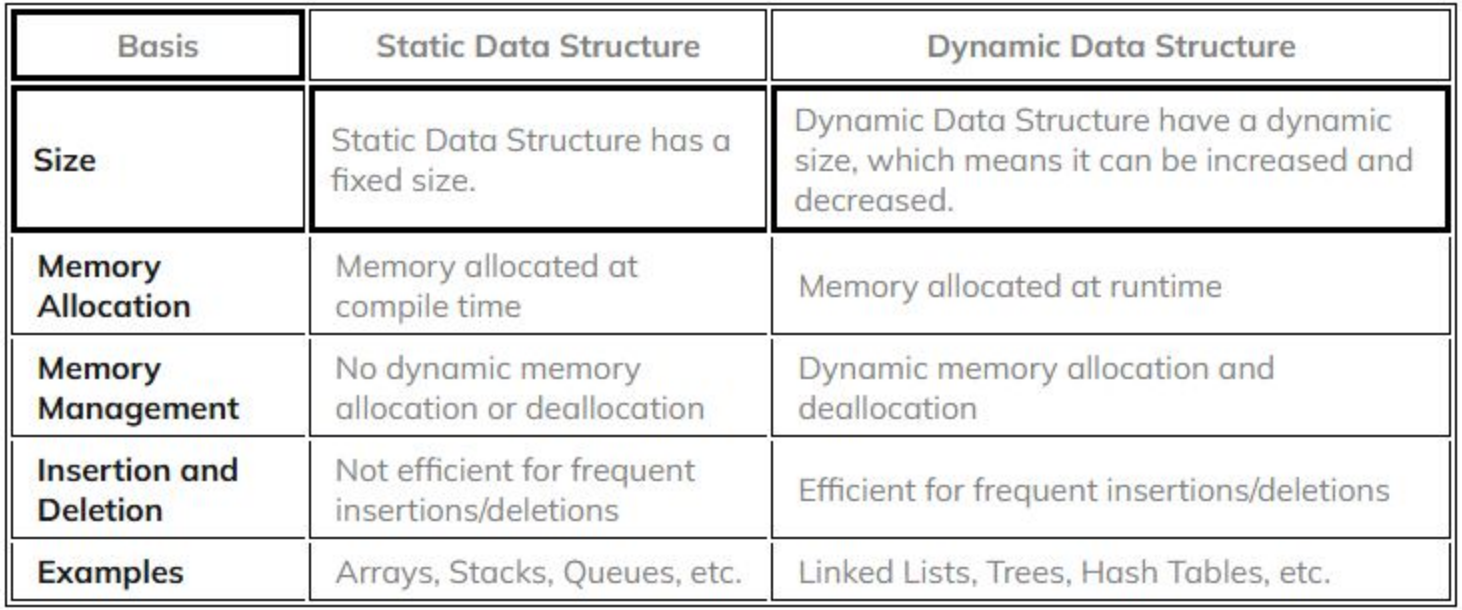
In Static data structure the size of the structure is fixed. The content of the data structure can be modified but without changing the memory space allocated to it.



What is Dynamic Data Structure?

In dynamic data structure, the size of the structure is not fixed and can be modified during the operations performed on it. Dynamic data structures are designed to facilitate change of data structures in the runtime.

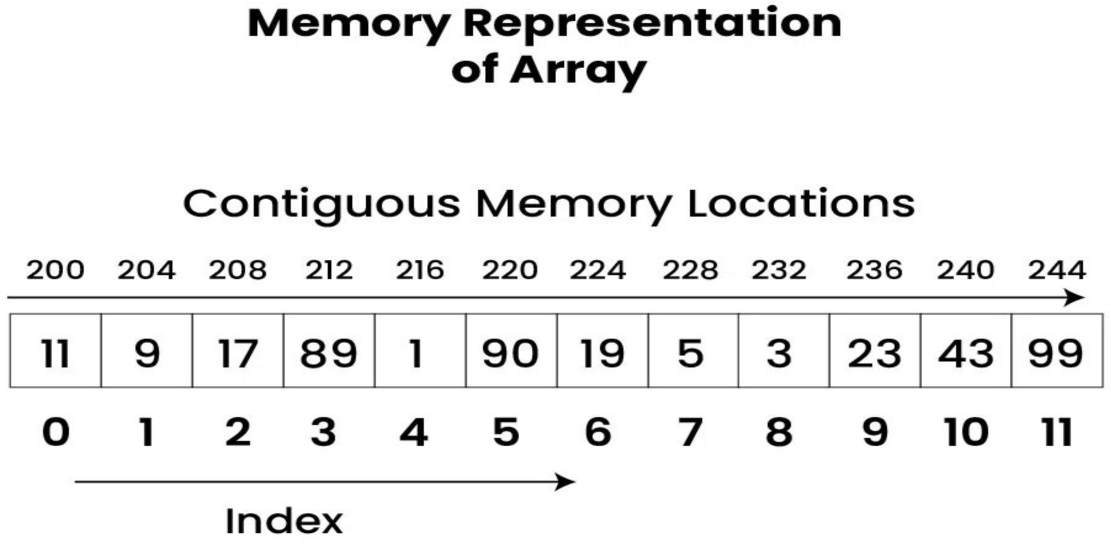




Basis	Static Data Structure	Dynamic Data Structure
Size	Static Data Structure has a fixed size.	Dynamic Data Structure have a dynamic size, which means it can be increased and decreased.
Memory Allocation	Memory allocated at compile time	Memory allocated at runtime
Memory Management	No dynamic memory allocation or deallocation	Dynamic memory allocation and deallocation
Insertion and Deletion	Not efficient for frequent insertions/deletions	Efficient for frequent insertions/deletions
Examples	Arrays, Stacks, Queues, etc.	Linked Lists, Trees, Hash Tables, etc.

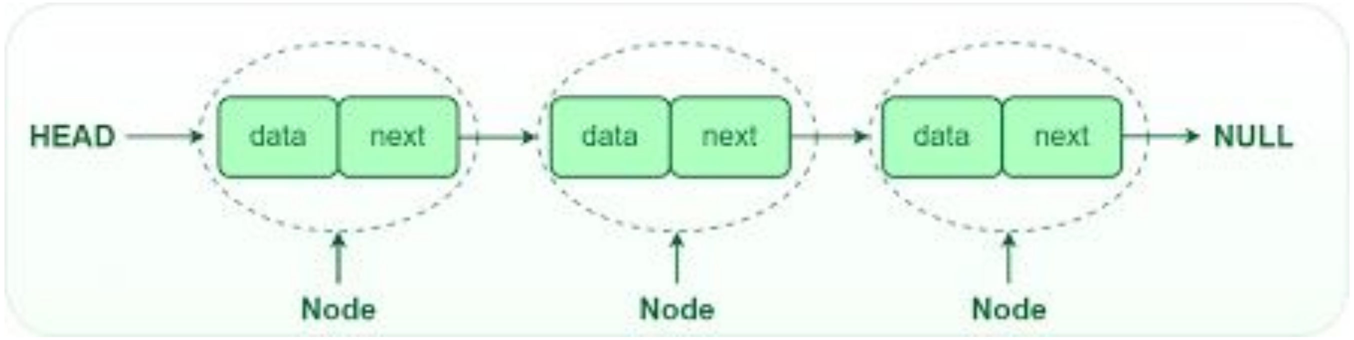
Array

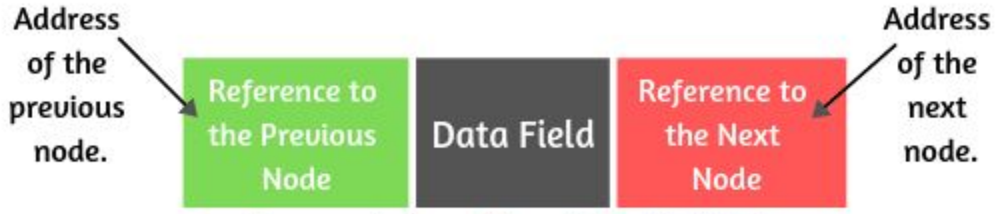
An array is a linear data structure where all elements are arranged sequentially. It is a collection of elements of the same data type stored at contiguous memory locations.

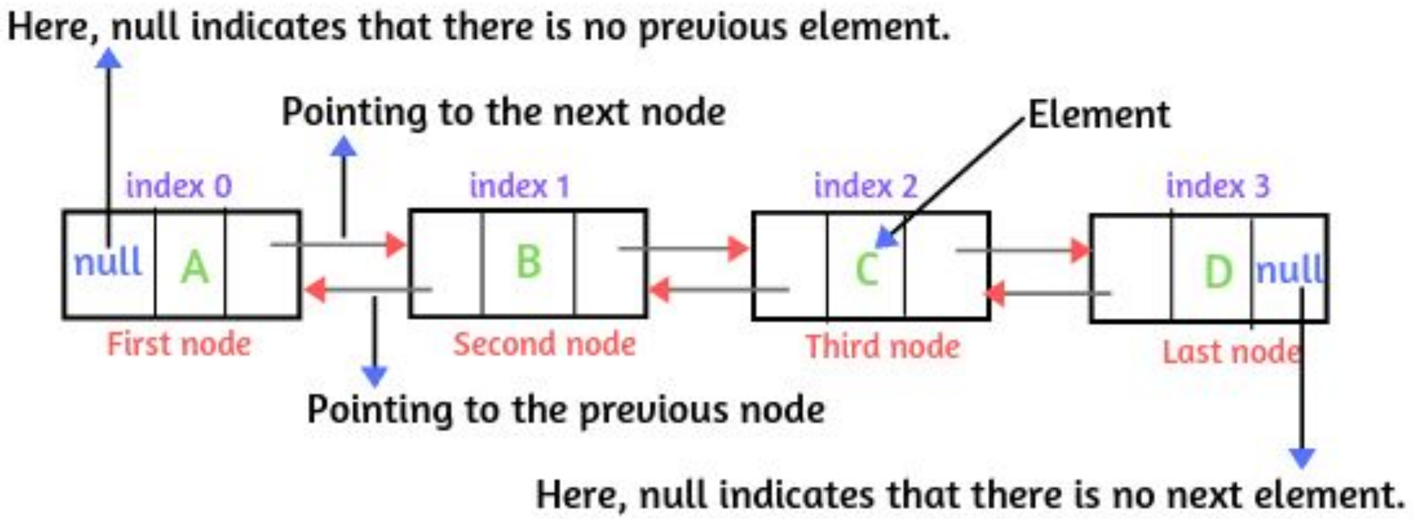


Linked List Data Structure

- A linked list is a dynamic linear data structure which can store a collection of "nodes" connected together via links i.e. pointers. Linked lists nodes are not stored at a contiguous location, rather they are linked using pointers to the different memory locations. A node consists of the data value and a pointer to the address of the next node within the linked list.
- It’s memory size can be allocated or de-allocated at run time based on the operation insertion or deletion, this helps in using system memory efficiently. Linked lists can be used to implement various data structures like a stack, queue, graph, hash maps, etc.

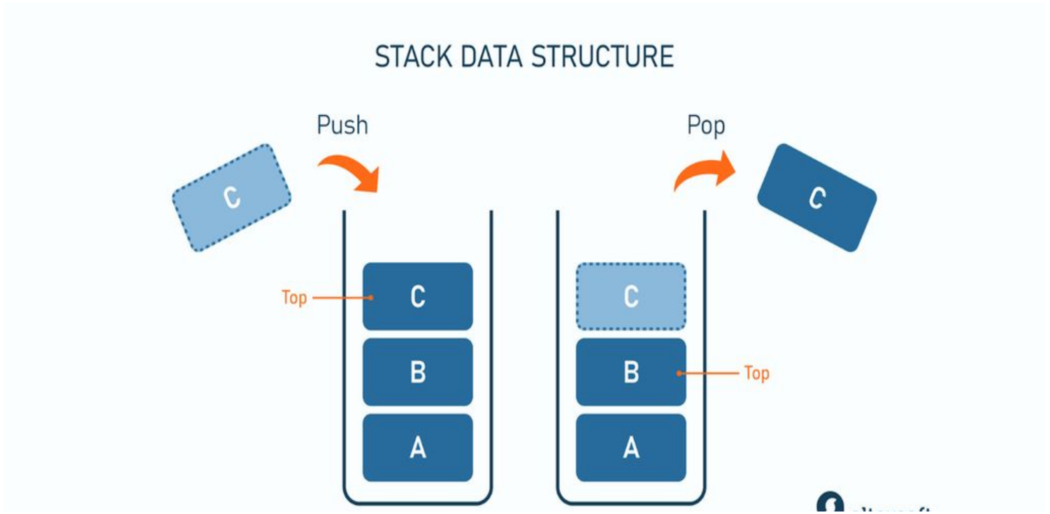


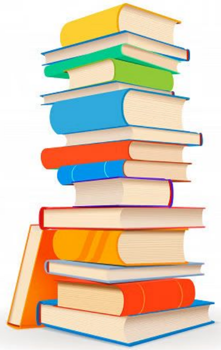

Address of the previous node.
Reference to the Previous Node
Data Field
Reference to the Next Node
Address of the next node.


Here, null indicates that there is no previous element.
Pointing to the next node
Element
index 0
index 1
index 2
index 3
null
A
B
C
D
null
First node
Second node
Third node
Last node
Pointing to the previous node
Here, null indicates that there is no next element.

Stack Data Structure

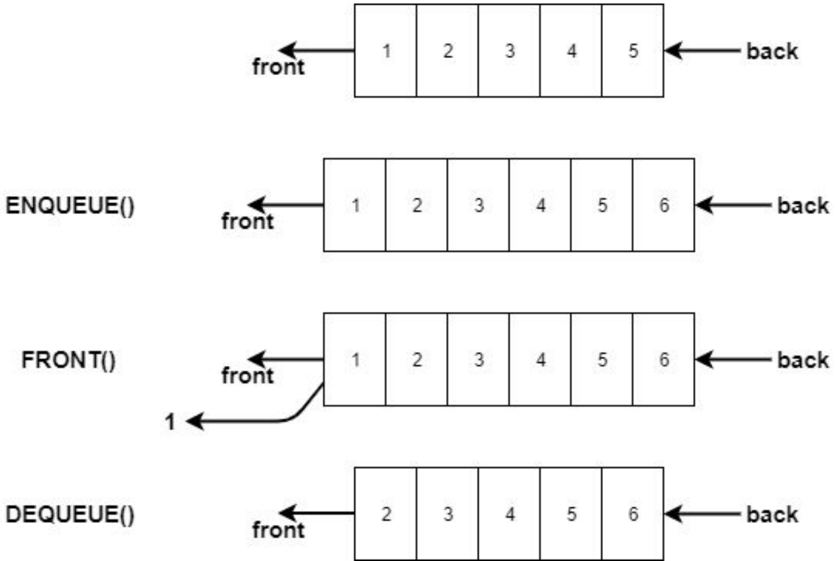
Stack is a linear data structure that follows **LIFO (Last In First Out) Principle,** the last element inserted is the first to be popped out. It means both insertion and deletion operations happen at one end only.





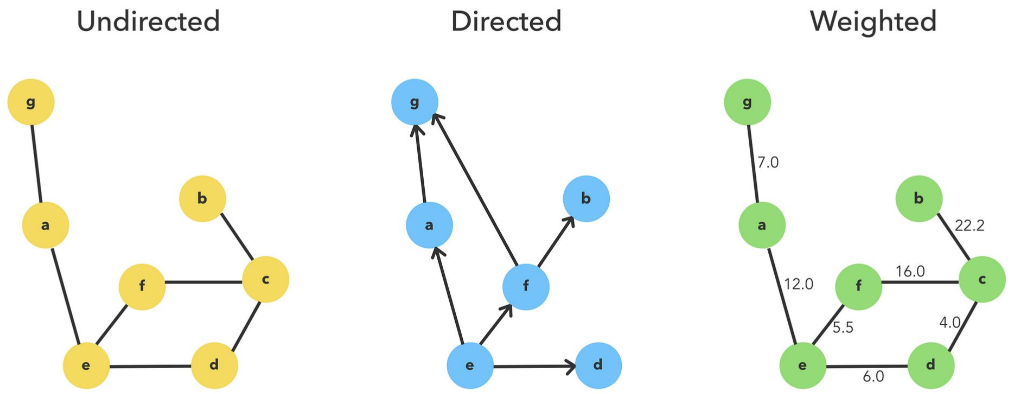
Queue Data Structure

- A queue is a useful data structure in programming. It is similar to the ticket queue outside a cinema hall, where the first person entering the queue is the first person who gets the ticket.
- Queue follows the **First In First Out (FIFO)** rule: the item that goes in first is the item that comes out first.

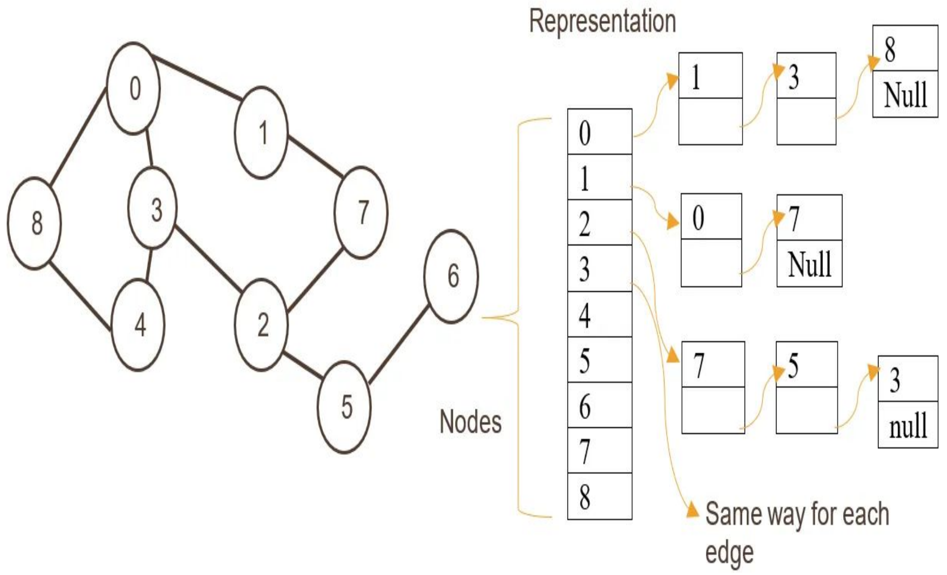


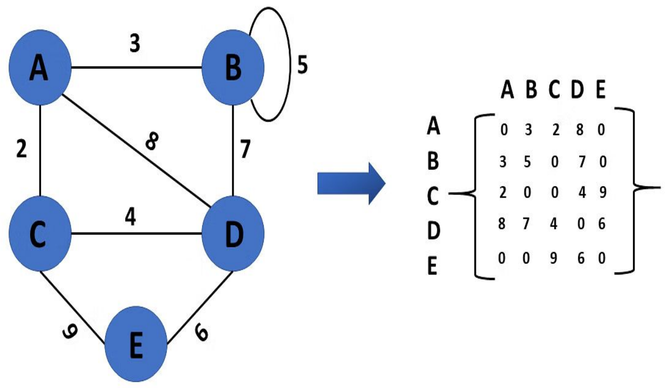
Graphs in Data Structure

A graph is a non-linear kind of data structure made up of nodes or vertices and edges. The edges connect any two nodes in the graph, and the nodes are also known as vertices.



Graph Representation

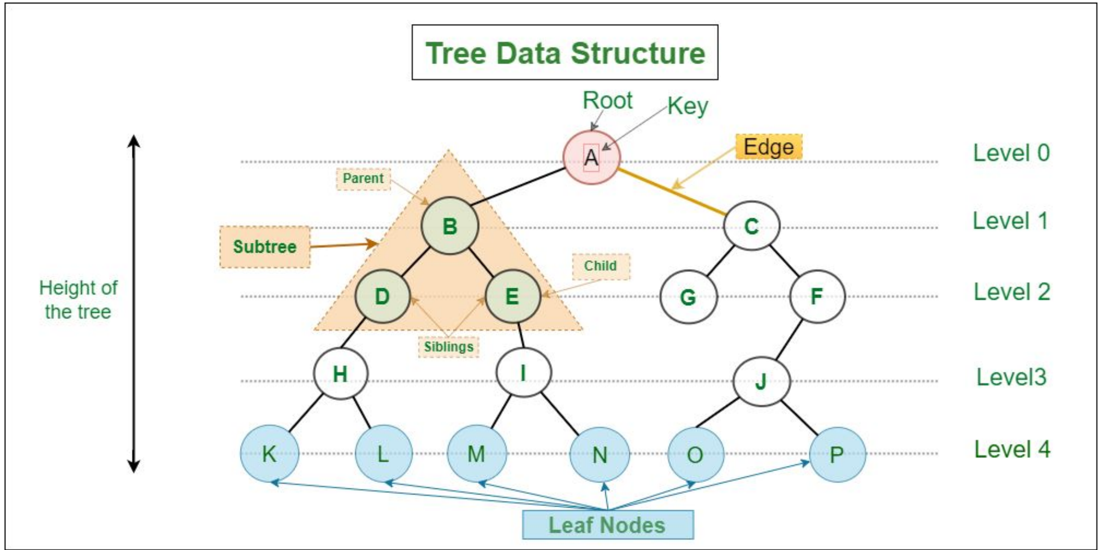




Tree Data Structure

Tree data structure is a hierarchical structure that is used to represent and organize data in the form of parent child relationship. The following are some real world situations which are naturally a tree.

- Folder structure in an operating system.
- Tag structure in an HTML (root tag the as html tag) or XML document.



References

[1] https://www.geeksforgeeks.org/static-data-structure-vs-dynamic-data-structure/