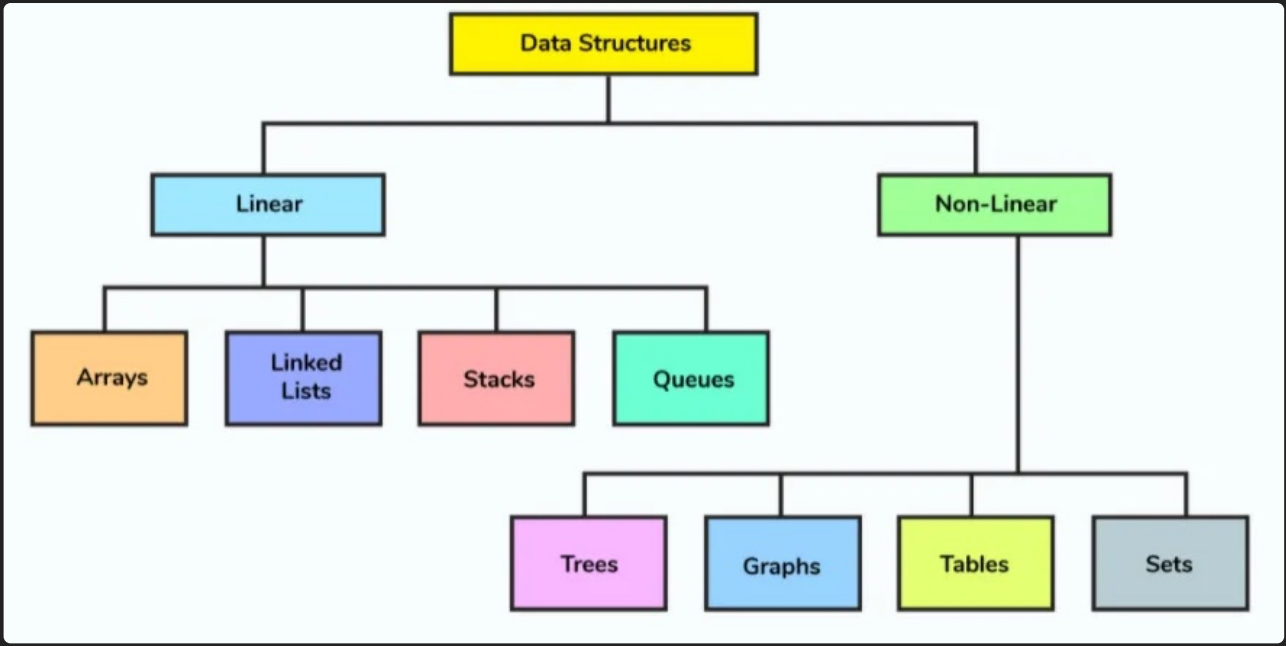
What is a Data Structure?

A way or organizing information in some fashion to be accessed, queried, or updated efficiently to benefit some task



Why Data Structures?

Lead to efficient algorithms

Helps us manage and organize data

Leads to cleaner code and makes our code easier to understand

Abstract Data Type

An abstraction of a data structure which only provides the interface or the underlying underlying understanding of how a data structure must act and the rules it must adhere to

Linked List

- Data Structure: Singly Linked List, Doubly Linked List, Circular Linked List

Stack

- Data Structure: Array, Linked List
- ADT Description: A stack is a

Queue

- Data Structure: Array, Linked List, Circular Buffer
- ADT Description: A queue is a

Deque

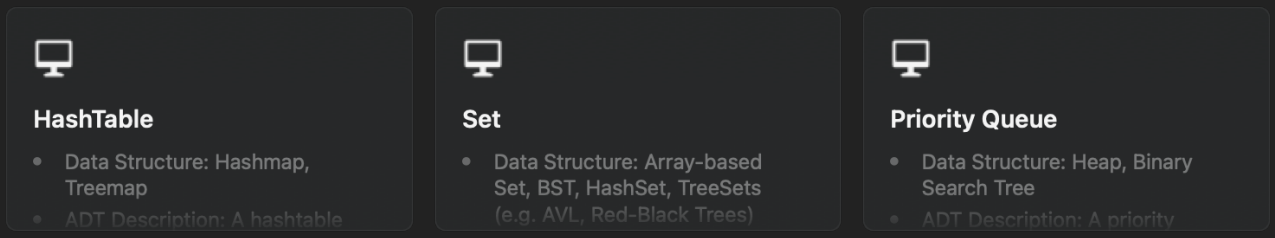
- Data Structure: Array, Linked List
- ADT Description: A deque is a

Tree

- Data Structure: Binary Tree, B-Tree, AVL Tree, Red-Black Tree

Graph

- Data Structure: Adjacency List, Adjacency Matrix
- ADT Description: A graph is a



↑ Abstract Data Structures Introduction

Linked List

- Data Structure: Singly Linked List, Doubly Linked List, Circular Linked List
- ADT Description: A linked list is a linear data structure consisting of a sequence of nodes, where each node contains a data element and a reference (link) to the next node. It allows efficient insertion and deletion at any position but requires sequential traversal for accessing elements.

↑ Abstract Data Structures Introduction

Stack

- Data Structure: Array, Linked List
- ADT Description: A stack is a linear data structure that follows the Last-In-First-Out (LIFO) principle. It has two main operations: push (inserting an element at the top) and pop (removing the top element).

↑ Abstract Data Structures Introduction

Queue

- Data Structure: Array, Linked List, Circular Buffer
- ADT Description: A queue is a linear data structure that follows the First-In-First-Out (FIFO) principle. It has two main operations: enqueue (inserting an element at the back) and dequeue (removing the front element).

↑ Abstract Data Structures Introduction

Deque

- Data Structure: Array, Linked List
- ADT Description: A deque is a linear data structure that allows insertion and deletion at both ends. It supports operations like insertFront, insertRear, deleteFront, deleteRear, getFront, getRear, etc. Deques can be implemented using arrays or linked list

↑ Abstract Data Structures Introduction

Tree

- Data Structure: Binary Tree, B-Tree, AVL Tree, Red-Black Tree
- ADT Description: A tree is a hierarchical data structure that consists of nodes connected by edges. It is a non-linear structure with a hierarchical relationship between elements. Trees are used for efficient searching, sorting, and organizing hierarchical data.

↑ Abstract Data Structures Introduction

Graph

- Data Structure: Adjacency List, Adjacency Matrix
- ADT Description: A graph is a non-linear data structure that consists of a set of nodes (vertices) and a set of edges that connect pairs of nodes. Graphs can represent various relationships and are used for modeling complex systems, networks, and relationships between objects.

↑ Abstract Data Structures Introduction

HashTable

- Data Structure: Hashmap, Treemap
- ADT Description: A hashtable is a data structure that provides efficient storage and retrieval of key-value pairs. It is based on an array-like structure, where keys are mapped to indices using a hash function. The key-value pairs are stored in buckets or slots within the array.

↑ Abstract Data Structures Introduction

Set

- Data Structure: Array-based Set, BST, HashSet, TreeSets (e.g. AVL, Red-Black Trees)
- ADT Description: A set is an unordered collection of unique elements. It does not allow duplicate values, and the order of elements is not significant. Sets provide efficient membership testing, insertion, and deletion operations.

↑ Abstract Data Structures Introduction

Priority Queue

- Data Structure: Heap, Binary Search Tree

- ADT Description: A priority queue is an abstract data type that allows elements to be inserted with a priority and supports efficient retrieval of the highest (or lowest) priority element. It has operations like enqueue (inserting an element with a priority) and dequeue (retrieving the element with the highest priority).