Data Structures Concepts with Python Implementation

Introduction to Data Structures

Data structures are fundamental concepts in computer science used to store, manage, and organize data efficiently.

They define the way data is stored, accessed, and modified. Different data structures are optimized for different tasks, and choosing

the right data structure is crucial for the efficiency of a program. In this document, we will cover the key data structures such as

Arrays, Linked Lists, Stacks, Queues, Trees, Graphs, Hash Tables, and more.

1. Arrays

Arrays are linear data structures that store elements in contiguous memory locations. Each element is identified by an index,

which starts from 0. Arrays are useful for storing a collection of similar items.

Operations:

- Access: O(1)

- Search: O(n) (linear search) or O(log n) (binary search for sorted arrays)

- Insertion: O(n)

- Deletion: O(n)

Example Use: Storing a list of integers, characters, or objects.

Implementation in Python:

In Python, arrays are implemented using the `list` data type.

2. Linked Lists

A linked list is a linear data structure where elements are stored in nodes, and each node points to the next node

in the list. Unlike arrays, linked lists do not store elements in contiguous memory locations. Instead, they consist of

nodes that

are linked together by pointers.

Operations:

- Access: O(n)

- Insertion: O(1) at the head, O(n) elsewhere

- Deletion: O(1) at the head, O(n) elsewhere

Types:

- Singly Linked List

- Doubly Linked List

- Circular Linked List

Implementation in Python:

Linked lists can be implemented using custom classes and nodes in Python.

3. Stacks

A stack is a linear data structure that follows the Last In First Out (LIFO) principle. Elements are added to the top of the stack

and removed from the top as well. Stacks are useful for tasks like backtracking, function calls, and undo mechanisms.

Operations:

- Push (Insertion): O(1)

- Pop (Removal): O(1)

- Peek (Access the top element): O(1)

Implementation in Python:

Stacks can be implemented using Python lists or the `collections.deque` class.

4. Queues

A queue is a linear data structure that follows the First In First Out (FIFO) principle. Elements are added at the rear and removed

from the front. Queues are used in scheduling tasks, buffering, and real-time processing.

Operations:

- Enqueue (Insertion): O(1)

- Dequeue (Removal): O(1)

- Peek (Access the front element): O(1)

Types:

- Simple Queue

- Circular Queue

- Priority Queue

Implementation in Python:

Queues can be implemented using the `collections.deque` class in Python.

5. Trees

Trees are hierarchical data structures where each node has a value and pointers to its child nodes. The top node is called the root,

and nodes without children are called leaves. Common types of trees include Binary Trees, Binary Search Trees (BST), AVL Trees, and more.

Operations (Binary Tree):

- Insertion: O(log n)

- Deletion: O(log n)

- Search: O(log n)

Types:

- Binary Tree

- Binary Search Tree (BST)

- AVL Tree (Balanced Tree)

- Heap

Implementation in Python:

Trees can be implemented using classes and pointers.

6. Graphs

Graphs are collections of nodes (vertices) and edges, where the edges represent relationships between the vertices. Graphs can be

directed or undirected, weighted or unweighted. Graphs are used in networking, social networks, search algorithms, and more.

Operations:

- Breadth-First Search (BFS): O(V + E)

- Depth-First Search (DFS): O(V + E)

Types:

- Directed Graphs

- Undirected Graphs

- Weighted Graphs

Implementation in Python:

Graphs can be implemented using dictionaries, adjacency lists, or adjacency matrices.

7. Hash Tables

Hash tables are data structures that map keys to values using a hash function. Hashing allows for efficient data retrieval in constant time O(1). Collisions (when two keys hash to the same value) are handled using techniques like chaining or open addressing.

Operations:

- Insertion: O(1)
- Deletion: O(1)
- Search: O(1)

Implementation in Python:

Hash tables are implemented as dictionaries (`dict`) in Python.

Python Implementation of Data Structures

At the end of this document, we will provide Python implementations of the above data structures,

including Arrays, Linked Lists, Stacks, Queues, Trees, Graphs, and Hash Tables.