A linked list is a linear data structure where elements, called nodes, are connected using pointers. Unlike arrays, linked lists do not store elements in contiguous memory locations, allowing for efficient insertion and deletion operations.

Key Components of a Linked List:

1. Node: The basic building block of a linked list, containing:
    - Data: The value or data that the node stores.
    - Pointer (or Reference): A link to the next node in the list.
2. Head: The first node in the linked list. It is used as the starting point to traverse the list.
3. Tail: The last node in the list, which typically points to null (or None in some programming languages) to indicate the end of the list.

Types of Linked Lists:

1. Singly Linked List: Each node contains data and a pointer to the next node. It can be traversed in only one direction.
2. Doubly Linked List: Each node contains data, a pointer to the next node, and a pointer to the previous node. It allows traversal in both directions.
3. Circular Linked List: In this variation, the last node points back to the first node, forming a circle. It can be singly or doubly linked.

Basic Operations:

1. Insertion: Adding a new node to the list (e.g., at the beginning, end, or a specific position).
2. Deletion: Removing a node from the list.
3. Traversal: Accessing each node in the list sequentially to perform some operation, such as searching for a value.
4. Searching: Finding a node that contains a specific value.