Aror University of Art, Architecture, Design and Heritage
SUKKUR,Sindh
Department of Multimedia and Gaming
Course: Data Structures CSC-221 (Practical)
Instructor: Engr. Fatima Jaffar

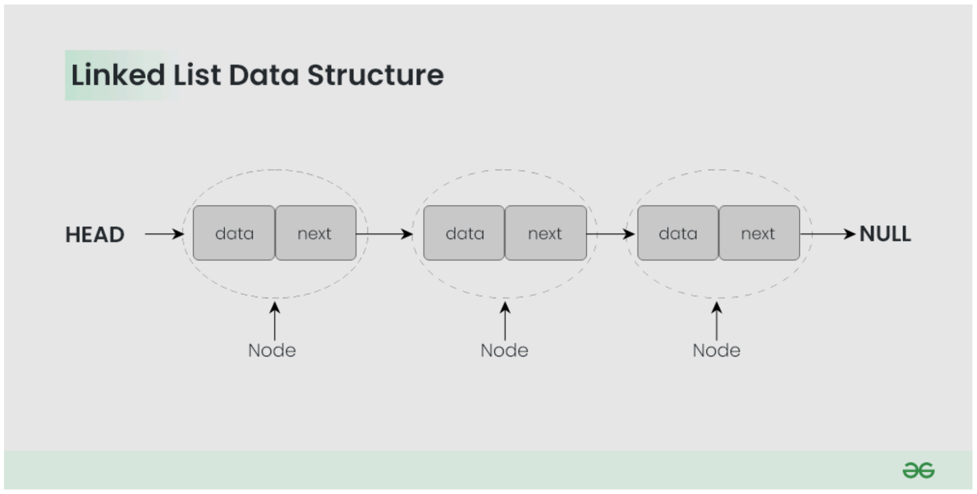
Lab No. 05

Objective: Understanding Singly linked List -Part 2

Name:				**Roll Number:**	
Score:		**Signature:**		**Date:**	18/09/2024

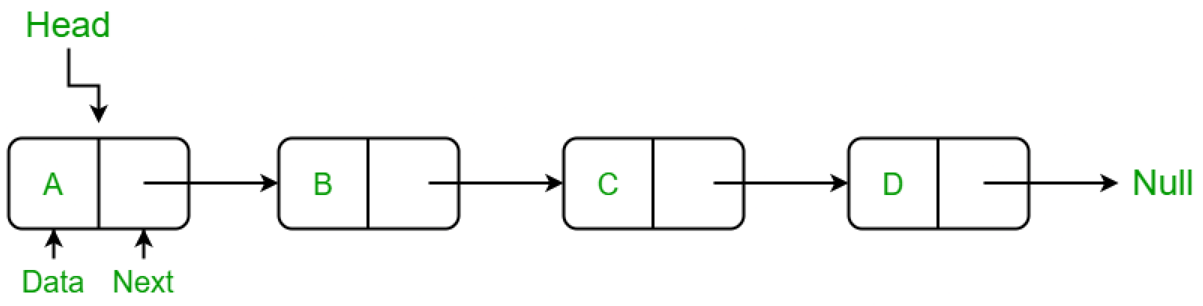
Introduction:

A linked list is a linear data structure, in which the elements are not stored at contiguous memory locations. The elements in a linked list are linked using pointers as shown in the below image:



Singly Linked List:

A singly linked list is a linear data structure in which the elements are not stored in contiguous memory locations and each element is connected only to its next element using a pointer.



Linked List Operations:

There are various linked list operations that allow us to perform different actions on linked lists. For example, the insertion operation adds a new element to the linked list.

Here's a list of basic linked list operations that we will cover in this article.

Insertion - adds a new element to the linked list

Traversal - access each element of the linked list

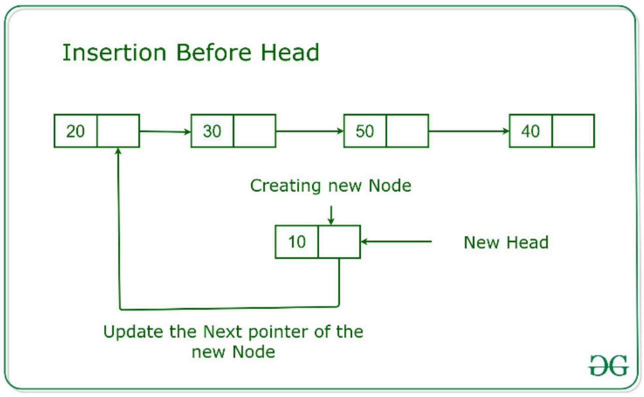
Deletion - removes the existing elements

Search - find a node in the linked list

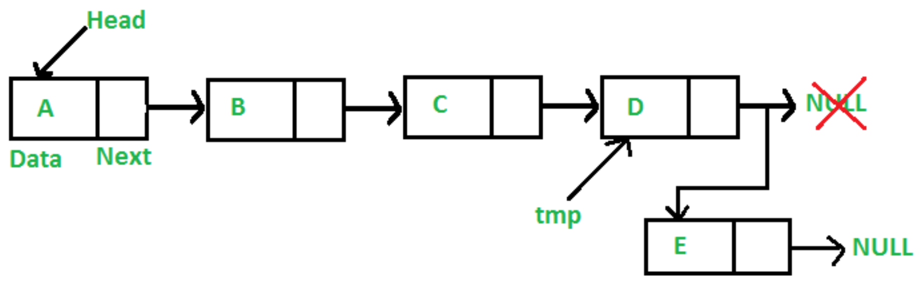
Sort - sort the nodes of the linked list.

1. Insertion:

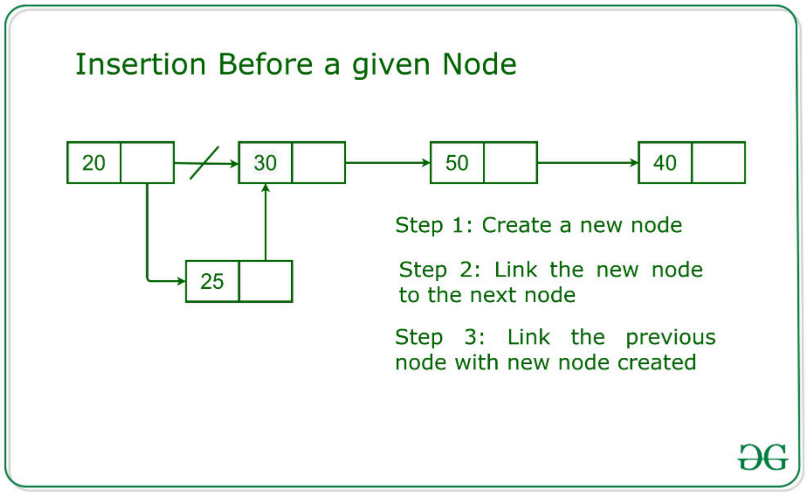
1.1. Insert first node



1.2. Insert Last node:



1.3. Insert after a specific node


Insertion Before a given Node
20
30
50
40
25
Step 1: Create a new node
Step 2: Link the new node to the next node
Step 3: Link the previous node with new node created

Implementation of Linked list in java

```
public class LinkedList{
Node Head;
}
public class Node{
Node next;
int data;
public Node(int val, Node next){
val=this.data;
next=this.next;
} }
```

Lab Tasks

1. Write a program to count the total number of nodes in a single linked list.
2. Traverse the linked list to find and display the maximum and minimum values stored in the nodes.
3. Write a program to find the sum of all the data values in the nodes of the linked list.
4. Implement a function to calculate and display the length of the linked list.
5. Swap the data of two nodes given their positions (e.g., swap the 2nd and 4th nodes).
6. Write a program to find and display the second last node of the linked list.
7. Implement a function to check if the linked list is empty and print an appropriate message.
8. Write a program to concatenate two single linked lists into one.
9. Calculate and display the average of the data values in the nodes of the linked list.
10. Implement a function to remove all nodes that contain a specific value from the linked list.
11. Write a program to print every alternate node in the linked list, starting from the first node.
12. Traverse the list and display the first node that contains an even number.
13. Implement a function that counts and displays how many times a specific value appears in the linked list.