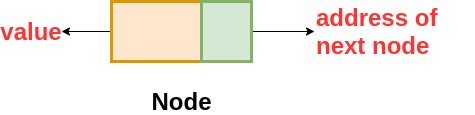
[**Visit My Github Profile**](https://github.com/panchalbhavya2210)

**Q - 1: EXPLAIN SINGLING LINKED LIST & OPERATION WE CAN PERFORM IN SINGLING LINKED LIST.**

ANS -> **Singly Linked List**

A Singly-linked list is a collection of nodes linked together in a sequential way where each node of the singly linked list contains a data field and an address field that contains the reference of the next node.

The structure of the node in the Singly Linked List is



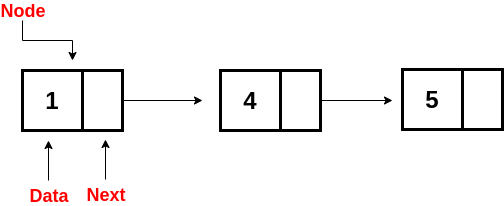
*class Node {*

*int data // variable to store the data of the node*

*Node next // variable to store the address of the next node*

*}*

The nodes are connected to each other in this form where the value of the next variable of the last node is NULL i.e. next = NULL, which indicates the end of the linked list.



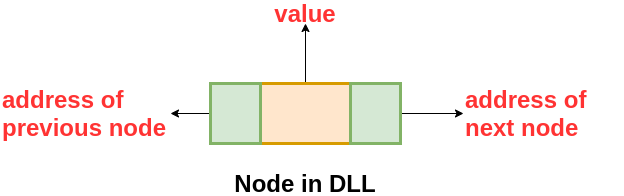
**Q - 2 : EXPLAIN DOUBLING LINKED LIST & OPERATION WE CAN PERFORM IN DOUBLING LINKED LIST.**

ANS ->

**Doubly Linked List**

A Doubly Linked List contains an extra memory to store the address of the previous node, together with the address of the next node and data which are there in the singly linked list. So, here we are storing the address of the next as well as the previous nodes.

The following is the structure of the node in the Doubly Linked List(DLL):



class DLLNode {

int val // variable to store the data of the node

DLLNode prev // variable to store the address of the previous node

DLLNode next // variable to store the address of the next node

}

The nodes are connected to each other in this form where the first node has prev = NULL and the last node has next = NULL.



**Advantages over Singly Linked List-**

* It can be traversed both forward and backward direction.
* The delete operation is more efficient if the node to be deleted is given. (Think! you will get the answer in the second half of this blog)
* The insert operation is more efficient if the node is given before which insertion should take place. (Think!)

**Disadvantages over Singly Linked List-**

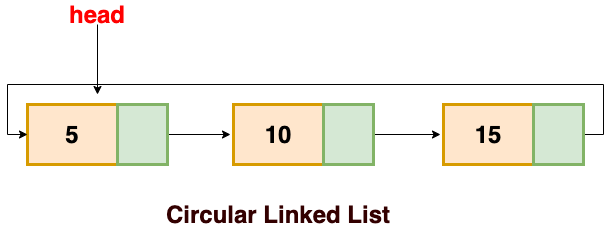
* It will require more space as each node has an extra memory to store the address of the previous node.
* The number of modification increase while doing various operations like insertion, deletion, etc.

**Q - 3 : EXPLAIN CIRCULAR LINKED LIST & OPERATION WE CAN PERFORM IN CIRCULAR LINKED LIST.**

ANS ->

**Circular Linked List**

A circular linked list is either a singly or doubly linked list in which there are no NULL values. Here, we can implement the Circular Linked List by making the use of Singly or Doubly Linked List. In the case of a singly linked list, the next of the last node contains the address of the first node and in case of a doubly-linked list, the next of last node contains the address of the first node and prev of the first node contains the address of the last node.



**Advantages of a Circular linked list**

* The list can be traversed from any node.
* Circular lists are the required data structure when we want a list to be accessed in a circle or loop.
* We can easily traverse to its previous node in a circular linked list, which is not possible in a singly linked list. (Think!)

**Disadvantages of Circular linked list**

* If not traversed carefully, then we could end up in an infinite loop because here we don't have any NULL value to stop the traversal.
* Operations in a circular linked list are complex as compared to a singly linked list and doubly linked list like reversing a circular linked list, etc.

**Basic Operations on Linked List**

* **Traversal**: To traverse all the nodes one after another.
* **Insertion**: To add a node at the given position.
* Deletion: To delete a node.
* **Searching**: To search an element(s) by value.
* **Updating**: To update a node.
* **Sorting**: To arrange nodes in a linked list in a specific order.
* **Merging**: To merge two linked lists into one.

**Q - 4: EXPLAIN INSERTION , DELETION IN A SINGLING LINKED LIST WITH EXAMPLE.**

ANS -> <https://www.javatpoint.com/insertion-in-singly-linked-list-at-beginning>

**Q - 5: EXPLAIN INSERTION , DELETION IN A DOUBLING LINKED LIST WITH EXAMPLE.**

ANS -> <https://www.tutorialspoint.com/data_structures_algorithms/doubly_linked_list_algorithm.htm>

**Q - 6: EXPLAIN INSERTION , DELETION IN A CIRCULAR LINKED LIST WITH EXAMPLE.**

ANS -> <https://www.geeksforgeeks.org/circular-linked-list/>