Types of Linkedlist

Types of Linked list

- Single linked list
- Double linked list
- Circular linked list

Single Linkedlist

Singly Linked List

- Singly linked lists contain nodes which have a **data** part as well as an **address part** i.e. next, which points to the next node in the sequence of nodes.

The operations we can perform on singly linked lists are

- **insertion**,
- **deletion**
- **traversal**.

Double Linked list

- Double linked list is a sequence of elements in which every element has links to its previous element and next element in the sequence.
- We add a pointer to the previous node in a doubly linked list. Thus, we can go in either direction: forward or backward.

Double linked list

Operations

In a double linked list, we perform the following operations...

- Insertion
- Deletion
- Display

Insertion

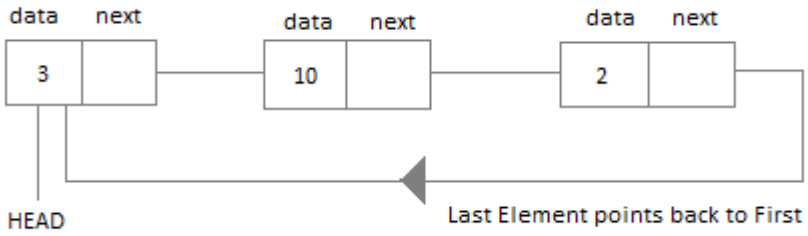
- In a double linked list, the insertion operation can be performed in three ways as follows...
- Inserting At Beginning of the list
- Inserting At End of the list
- Inserting At Specific location in the list

Deletion

- In a double linked list, the deletion operation can be performed in three ways as follows...
- Deleting from Beginning of the list
- Deleting from End of the list
- Deleting a Specific Node

Circular Linked List

- In circular linked list the last node of the list holds the address of the first node hence forming a circular chain.



Operation in Circular Linkedlist

- Insertion at the Beginning
- Insertion at the End
- Searching for an Element in the List
- Deleting a Node from the List

Example

```
#include<stdio.h>
#include<stdlib.h>
 struct Node
{
  int data;
  struct Node *next;
};
 // This function prints contents of linked list starting from
// the given node
void printList(struct Node *n)
{
```

```
 while (n != NULL)
  {
     printf(" %d ", n->data);
     n = n->next;
  }
}
int main()
{
  struct Node* head = NULL;
  struct Node* second = NULL;
  struct Node* third = NULL;
// allocate 3 nodes in the heap
  head  = (struct Node*)malloc(sizeof(struct Node));
  second = (struct Node*)malloc(sizeof(struct Node));
  third  = (struct Node*)malloc(sizeof(struct Node));

  head->data = 1; //assign data in first node
  head->next = second; // Link first node with second
```

```
second->data = 2; //assign data to second node
second->next = third;
third->data = 3; //assign data to third node
third->next = NULL;
printList(head);
return 0;
}
```