**Question NO:06**

#include<iostream>

using namespace std;

class Node{

public:

int data;

Node \*next;

Node(){

data = 0;

next = NULL;

}

Node(int d){

this->data = d;

this->next = NULL;

}

};

class LinkedList{

Node \*head;

public:

LinkedList(){

this->head = NULL;

}

//insertation faciltators

void insertAtStart(int d){

Node \*newNode = new Node(d);

newNode->next = head;

head =newNode;

}

void insertAtLast(int d){

Node \*newNode = new Node(d);

Node \* temp = head;

while(temp->next!= NULL){

temp = temp->next;

}

temp->next = newNode;

}

void insertAtAnyPosition(int d, int po){

if(po < 1){

cout<<"Invalid Positoin\n";

}

else if( po == 1){

insertAtStart(d);

}

else{

Node \*newNode = new Node(d);

Node \*temp = head;

for(int i = 1 ; i<po-1 ;i++){

if(temp == NULL){

cout<<"Invalid Postion";

break;

}

temp = temp->next;

}

newNode->next = temp->next;

temp->next = newNode;

}

}

//Deletion facalitators

void delete\_at\_Start(){

if(head!= NULL){

Node\* temp = head;

head = head->next;

delete temp;

temp = NULL;

}

}

void delete\_at\_position(int pos){

if(head!= NULL){

if(pos<1){

cout<<"Invalid Position\n";

}

else if(pos == 1){

delete\_at\_Start();

}

else {

Node\* temp = head;

for( int i = 1; i<pos-1; i++){

temp = temp->next;

if(temp == NULL){

break;

}

}

Node\* temp2 = temp->next->next;

delete temp->next;

temp->next = temp2;

}

}

}

void delete\_lastNode(){

if(head!= NULL){

Node\* temp = head;

Node\* temp2 = NULL;

while(temp->next!= NULL){

temp2 = temp;

temp = temp->next;

}

temp2->next = NULL;

delete temp;

temp = NULL;

}

}

void delete\_Center\_Node(){

if(head!= NULL){

Node\* temp = head;

int count = 1;

while (temp!= NULL)

{

count++;

temp = temp->next;

}

temp = head;

Node\* temp2 = NULL;

for(int i = 1; i<count/2; i++){

temp2 = temp;

temp= temp->next;

}

temp2->next = temp->next;

delete temp;

}

}

void delete\_a\_value (int v){

if(head!= NULL){

Node\* temp = head;

Node\* temp2 = NULL;

while(temp != NULL && temp->data != v){

temp2 =temp;

temp = temp->next;

}

temp2->next = temp->next;

delete temp;

}

}

void displayLinknedList(){

Node \*temp = head;

while(temp != NULL){

cout<<temp->data;

if(temp->next != NULL){

cout<<",";

}

temp = temp->next;

}

cout<<endl;

}

void displayFirstNode(){

cout<<head->data<<endl;

}

void displaylastNode(){

Node \*temp = head;

while(temp->next != NULL){

temp = temp->next;

}

cout<<temp->data<<endl;

}

void displayCenterNode(){

Node \*temp = head;

int count = 0;

while(temp != NULL){

temp= temp->next;

count++;

}

temp = head;

for(int i = 1 ; i<count; i++){

temp = temp->next;

if(i == count/2){

cout<<temp->data<<endl;

break;

}

}

}

};

int main(){

LinkedList list;

list.insertAtStart(9);

list.insertAtStart(93);

list.insertAtLast(0);

list.insertAtLast(88);

list.insertAtAnyPosition(8,2);

list.insertAtAnyPosition(77,3);

list.insertAtAnyPosition(18,2);

cout<<"Before LinkedList : ";

list.displayLinknedList();

cout<<"First Linked List Node : ";

list.displayFirstNode();

cout<<"Last LinkedList Node : ";

list.displaylastNode();

cout<<"Center linkedList : ";

list.displayCenterNode();

cout<<"Deleteing the Last Node.\n";

list.delete\_lastNode();

cout<<"Last Linked Node :";

list.displaylastNode();

cout<<"Deleteing the First Node.\n";

list.delete\_at\_Start();

cout<<"First Linked List Node : ";

list.displayFirstNode();

cout<<"Deleteing the Center Node.\n";

list.delete\_Center\_Node();

cout<<" Center linkedList : ";

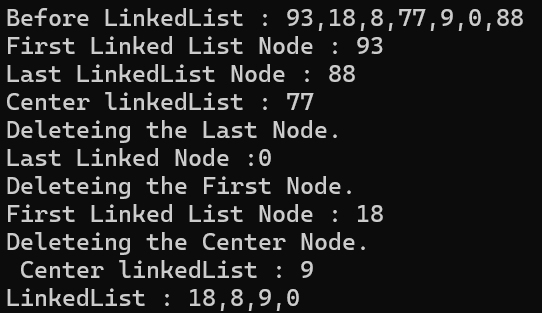
list.displayCenterNode();

cout<<"LinkedList : ";

list.displayLinknedList();

}

**OUTPUT**

****

**Explanation**

This code creates a linked list that can add and remove nodes. You can insert nodes at the start, end, or a specific position. It also lets you delete nodes from the start, end, center, or by value. The code includes methods to show the first, last, and center nodes, as well as the entire list. In the main function, nodes are added and removed to demonstrate how the linked list works.