Name- Abhinav Kumar
PRN- 21070126006
Branch- AIML-A1
Lab Assignment- Doubly LinkList

```
#include<iostream>
using namespace std;

// Defininig structure for node
struct node
{
    int data;
    struct node *next;
    struct node *prev;
};

// Traverse the list
void traverse (struct node *head)
{
    struct node *temp=head;
    while (temp!=NULL)
    {
        cout<<temp->data<<"->";
        temp=temp->next;
    }
}

// Insert at the beginning of the list
void insert_beg (struct node *head)
{
    int n;
    cout<<"Enter data of new node: ";
    cin>>n;
    struct node *newnode=NULL;
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data=n;
    newnode->next=NULL;
    newnode->next=head;
    newnode->prev=NULL;
    head->prev=newnode;
    head=newnode;
    traverse(head);
}

// Insert at the end of the list
void insert_end(struct node *head)
{
    int n;
    cout<<"Enter data of new node: ";
    cin>>n;
```

```
    struct node *newnode=NULL;
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data=n;
    newnode->next=NULL;
    newnode->prev=NULL;
    struct node *temp=head;
    while(temp->next != NULL)
    {
        temp=temp->next;
    }
    temp->next=newnode;
    newnode->prev=temp;
    traverse(head);
}

// Insert at a position in list
void insert_pos(struct node *head)
{
    int n, position;
    cout<<"Enter data of new node: ";
    cin>>n;
    cout<<"Enter position to enter: ";
    cin>>position;
    struct node *newnode=NULL;
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data=n;
    newnode->next=NULL;
    newnode->prev=NULL;

    struct node *temp=head;
    for(int i=1; i<position-1; i++)
    {
        temp=temp->next;
    }
    newnode->next=temp->next;
    newnode->prev=temp;
    temp->next=newnode;
    temp->next->prev=newnode;
    traverse(head);
}

// Delete at beginning of list
void delete_beg(struct node *head)
{
    struct node *temp=head;
    head=head->next;
    head->prev=NULL;
    free(temp);
    traverse(head);
}
```

```
// Delete at end of list
void delete_end(struct node *head)
{
    struct node *temp=head;
    while(temp->next != NULL)
    {
        temp=temp->next;
    }
    temp->prev->next=NULL;
    free(temp);
    traverse(head);
}

// Delete at a position in list
void delete_pos(struct node *head)
{
    struct node *temp=head;
    int position;
    cout<<"Enter position to delete: ";
    cin>>position;
    for (int i=1; i<position; i++)
    {
        temp=temp->next;
    }
    temp->prev->next=temp->next;
    temp->next->prev=temp->prev;
    free(temp);
    traverse(head);
}

// Concatenate two lists
void concatenate(struct node *head1)
{
    struct node *head2=NULL;
    struct node *second2=NULL;
    head2=(struct node*)malloc(sizeof(struct node));
    second2=(struct node*)malloc(sizeof(struct node));

    head2->data=10;
    head2->next=second2;
    head2->prev=NULL;
    second2->data=20;
    second2->next=NULL;
    second2->prev=head2;

    struct node *temp=head1;
    while(temp->next != NULL)
    {
        temp=temp->next;
```

```
    }
    temp->next=head2;
    head2->prev=temp;
    traverse(head1);
}

void menu(struct node* head)
{
    //Creating a menu
    int choice;
    cout<<"\n\n1. Traverse the list"<<endl;
    cout<<"2. Insert at the beginning"<<endl;
    cout<<"3. Insert at the end"<<endl;
    cout<<"4. Insert at a position"<<endl;
    cout<<"5. Delete at the beginning"<<endl;
    cout<<"6. Delete at the end"<<endl;
    cout<<"7. Delete at a position"<<endl;
    cout<<"8. Concatenate two lists"<<endl;
    cout<<"9. Exit"<<endl;

    //Taking input from user
    cout<<"Enter your choice: ";
    cin>>choice;

    if (choice==1)
    {
        traverse(head);
    }
    else if (choice==2)
    {
        insert_beg(head);
    }
    else if (choice==3)
    {
        insert_end(head);
    }
    else if (choice==4)
    {
        insert_pos(head);
    }
    else if (choice==5)
    {
        delete_beg(head);
    }
    else if (choice==6)
    {
        delete_end(head);
    }
    else if (choice==7)
    {
```

```
        delete_pos(head);
    }
    else if (choice==8)
    {
        concatenate(head);
    }
    else if (choice==9)
    {
        exit(0);
    }
    else
    {
        cout<<"Invalid choice"<<endl;
    }
    menu(head);
}

// Main function
int main()
{
    struct node *head=NULL;
    struct node *second=NULL;
    struct node *third=NULL;

    // Allocating memory for nodes
    head=(struct node*)malloc(sizeof(struct node));
    second=(struct node*)malloc(sizeof(struct node));
    third=(struct node*)malloc(sizeof(struct node));

    // Assigning values to nodes
    head->prev=NULL;
    head->data=45;
    head->next=second;
    second->prev=head;
    second->data=80;
    second->next=third;
    third->prev=second;
    third->data=95;
    third->next=NULL;

    menu(head);
    return 0;
}
```