WAP to Implement doubly link list with primitive operations a) Create a doubly linked list. b) Insert a new node to the left of the node. c) Delete the node based on a specific value

d) Display the contents of the list

Program:

#include <stdio.h> #include <stdlib.h> struct Node

{

int data;

struct Node\* prev; struct Node\* next;

};

void create(struct Node\*\* head, int data)

{

struct Node\* new\_node = (struct Node\*)malloc(sizeof(struct Node)); new\_node->data = data;

new\_node->prev = NULL; new\_node->next = NULL; if (\*head == NULL)

{

\*head = new\_node; return;

}

struct Node\* temp = \*head; while (temp->next != NULL)

{

temp = temp->next;

}

temp->next = new\_node; new\_node->prev = temp;

}

void insert\_left(struct Node\*\* head, int target\_data, int new\_data)

{

struct Node\* new\_node = (struct Node\*)malloc(sizeof(struct Node)); new\_node->data = new\_data;

struct Node\* temp = \*head; while (temp != NULL)

{

if (temp->data == target\_data)

{

new\_node->next = temp; new\_node->prev = temp->prev; if (temp->prev != NULL)

{

temp->prev->next = new\_node;

}

else

{

\*head = new\_node;

}

temp->prev = new\_node; return;

}

temp = temp->next;

}

printf("Node with data %d not found.\n", target\_data);

}

void delete\_node(struct Node\*\* head, int value)

{

struct Node\* temp = \*head; while (temp != NULL)

{

if (temp->data == value)

{

if (temp == \*head)

{

\*head = temp->next;

}

if (temp->prev != NULL)

{

temp->prev->next = temp->next;

}

if (temp->next != NULL)

{

temp->next->prev = temp->prev;

}

free(temp); return;

}

temp = temp->next;

}

printf("Node with data %d not found.\n", value);

}

void display(struct Node\* head)

{

if (head == NULL) { printf("The list is empty.\n"); return;

}

struct Node\* temp = head; while (temp != NULL)

{

printf("%d", temp->data); if (temp->next != NULL)

{

printf(" <-> ");

}

temp = temp->next;

}

printf("\n");

}

int main()

{

struct Node\* head = NULL;

int choice, data, target\_data, new\_data;

while (1)

{

printf("\nDoubly Linked List Operations:\n"); printf("1. Create a node\n");

printf("2. Insert node to the left of a specific node\n"); printf("3. Delete a node\n");

printf("4. Display the list\n"); printf("5. Exit\n"); printf("Enter your choice: "); scanf("%d", &choice);

switch (choice)

{

case 1:

printf("Enter the data for the node to create: ");

scanf("%d", &data); create(&head, data); break;

case 2:

printf("Enter the target node data before which to insert: "); scanf("%d", &target\_data);

printf("Enter the data for the new node to insert: "); scanf("%d", &new\_data);

insert\_left(&head, target\_data, new\_data); break;

case 3:

printf("Enter the data of the node to delete: "); scanf("%d", &data);

delete\_node(&head, data); break;

case 4:

printf("The current list is: "); display(head);

break;

case 5:

printf("Exiting...\n"); exit(0);

default:

printf("Invalid choice. Please try again.\n");

}

}

return 0;

}

