# **LAB 5**

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

#include <stdio.h>

#include <stdlib.h>

struct node

{

int data;

struct node \*prev;

struct node \*next;

};

struct node \*head = NULL, \*ptr, \*temp, \*newnode;

void create\_ll();

void insert\_before();

void delete\_node();

void display();

void create\_ll()

{

int value;

newnode = (struct node \*)malloc(sizeof(struct node));

printf("Enter the value to be inserted:");

scanf("%d", &value);

if (head == NULL)

{

newnode->data = value;

newnode->prev = NULL;

newnode->next = NULL;

head = newnode;

}

else

{

ptr = head;

while (ptr->next != NULL)

ptr = ptr->next;

newnode->data = value;

newnode->prev = ptr;

newnode->next = NULL;

ptr->next = newnode;

}

}

void insert\_before()

{

int value, num;

newnode = (struct node \*)malloc(sizeof(struct node));

printf("Enter the value to be inserted:");

scanf("%d", &value);

printf("Enter before which the value should be inserted:");

scanf("%d", &num);

newnode->data = value;

if (head == NULL || num == head->data)

{

newnode->prev = NULL;

newnode->next = head;

if (head != NULL)

{

head->prev = newnode;

}

head = newnode;

return;

}

ptr = head;

while (ptr->data != num)

ptr = ptr->next;

newnode->prev = ptr->prev;

newnode->next = ptr;

ptr->prev->next = newnode;

ptr->prev = newnode;

}

void delete\_node()

{

int value;

ptr = head;

printf("Enter the value to be deleted:");

scanf("%d", &value);

if (head == NULL)

{

printf("List is empty");

return;

}

if (value == head->data)

{

temp = head;

head = head->next;

head->prev = NULL;

free(temp);

return;

}

while(ptr->data!=value){

temp=ptr;

ptr=ptr->next;

}

temp->next=ptr->next;

ptr->next->prev=temp;

free(ptr);

if (ptr == NULL)

{

printf("Value not found in the list");

}

}

void display()

{

if (head == NULL)

{

printf("List is empty");

}

else

{

ptr = head;

while (ptr != NULL)

{

printf("%d ", ptr->data);

ptr = ptr->next;

}

}

}

int main()

{

int choice;

printf("1. create list 2. insert left 3. delete node 4. display 5. exit ");

while (1)

{

printf("\nEnter choice:");

scanf("%d", &choice);

switch (choice)

{

case 1:

create\_ll();

break;

case 2:

insert\_before();

break;

case 3:

delete\_node();

break;

case 4:

display();

break;

case 5:

exit(0);

break;

default:

printf("Invalid input!");

}

}

return 0;

}