# **Double linked list(27//11/24) - Insertion & Deletion**

### Program:

#include<stdio.h>

#include<stdlib.h>

struct node{

struct node \*prev;

int data;

struct node \*next;

};

struct node \*head = NULL;

void display();

struct node \*createnode(){

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

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

scanf("%d", &newnode->data);

newnode->prev = NULL;

newnode->next = NULL;

return newnode;

}

int main(){

int option;

while(1){

printf("ENTER \n1. INSERT \n2. DELETE \n3. DISPLAY \n4. EXIT\n");

scanf("%d",&option);

switch(option){

case 1:

insert();

break;

case 2:

delete();

break;

case 3:

display();

break;

case 4:

exit(0);

break;

default:

printf("INVALID ENTRY");

}

}

}

void insert(){

struct node \*newnode = createnode(), \*ptr = head;

int val;

printf("Enter the value before which insertion has to be done: ");

scanf("%d",&val);

if(head == NULL){

newnode->next = NULL;

newnode->prev = NULL;

head = newnode;

}else if(head->data == val){

newnode->next = head;

newnode->prev = NULL;

head->prev = newnode;

head = newnode;

}else {

while(ptr->data!=val)

ptr= ptr->next;

if (ptr == NULL) {

printf("Value not found for insertion.\n");

free(newnode);

}

newnode->next = ptr;

newnode->prev = ptr->prev;

ptr->prev->next = newnode;

ptr->prev = newnode;

}

}

int delete() {

int value;

printf("Enter value to delete: ");

scanf("%d", &value);

struct node \*ptr = head;

if (head == NULL) {

printf("List is empty\n");

return;

}

while (ptr != NULL && ptr->data != value) {

ptr = ptr->next;

}

if (ptr == NULL) {

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

return;

}

if (ptr == head) {

head = ptr->next;

if (head != NULL) {

head->prev = NULL;

}

} else if (ptr->next == NULL) {

ptr->prev->next = NULL;

} else {

ptr->prev->next = ptr->next;

ptr->next->prev = ptr->prev;

}

free(ptr);

return 0;

}

void display(){

struct node \*ptr = head;

if(head==NULL){

printf("LIST EMPTY\n");

}else{

while(ptr!=NULL){

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

ptr = ptr->next;

}

}

}

OUTPUT:







