## **LAB 2**

## OBJECTIVE: To implement Doubly Linked List and perform various operations in C programming.

## THEORY:

A doubly linked list is a data structure that consists of nodes that are linked together in both directions. Each node has three parts: data, a pointer to the next node, and a pointer to the previous node.



Operations to perform in single linked list are mention bellow:

1. Insert Operation
   1. Insert at beginning
   2. Insert at Position
   3. Insert at End
2. Delete Operation
   1. Delete at beginning
   2. Delete at position
   3. Delere at End
3. Traverse Operation
   1. Display items

## PROGRAM:

*#include<stdio.h>*

*#include<stdlib.h>*

*#include<conio.h>*

*// Doubly Linked List structure...*

*struct Node {*

*int data;*

*struct Node \*prev;*

*struct Node \*next;*

*}; typedef struct Node node;*

*node \*head = NULL;*

*// Function to create new node...*

*node\* createNewNode(int item) {*

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

*newnode->data = item;*

*newnode->prev = NULL;*

*newnode->next = NULL;*

*return newnode;*

*}*

*// Function to insert at beginning*

*void insertAtBeg(int item) {*

*node \*newnode = createNewNode(item);*

*if (head != NULL) {*

*newnode->next = head;*

*head->prev = newnode;*

*}*

*head = newnode;*

*}*

*// Function to insert at a specific position*

*void insertAtPos(int pos, int item) {*

*if (pos < 1) {*

*printf("\n\t\t!!! Invalid Position...\n");*

*return;*

*}*

*if (pos == 1) {*

*insertAtBeg(item);*

*return;*

*}*

*node \*temp = head;*

*node \*newnode = createNewNode(item);*

*for (int i = 1; i < pos - 1 && temp != NULL; i++)*

*temp = temp->next;*

*if (temp != NULL) {*

*newnode->next = temp->next;*

*if (temp->next != NULL)*

*temp->next->prev = newnode;*

*temp->next = newnode;*

*newnode->prev = temp;*

*} else {*

*printf("\n\t\t!!! Invalid Position...\n");*

*}*

*}*

*// Function to insert at the end*

*void insertAtEnd(int item) {*

*node \*newnode = createNewNode(item);*

*if (head == NULL) {*

*head = newnode;*

*return;*

*}*

*node \*temp = head;*

*while (temp->next != NULL)*

*temp = temp->next;*

*temp->next = newnode;*

*newnode->prev = temp;*

*}*

*// Function to delete at the beginning*

*void deleteAtBeg(){*

*if(head == NULL) printf("\n\t!!!Empty Node list...\n");*

*else {*

*head = head->next;*

*if (head != NULL)*

*head->prev = NULL;*

*}*

*}*

*// Function to delete at position*

*void deleteAtPos(int pos){*

*if (pos < 1) {*

*printf("\n\t\t!!! Invalid Position...\n");*

*return;*

*}*

*if (pos == 1) {*

*deleteAtBeg();*

*return;*

*}*

*node \*temp = head;*

*for (int i = 1; i < pos && temp != NULL; i++)*

*temp = temp->next;*

*if (temp != NULL) {*

*if (temp->prev != NULL)*

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

*if (temp->next != NULL)*

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

*} else {*

*printf("\n\t\t!!! Invalid Position...\n");*

*}*

*}*

*// Function to delete at end*

*void deleteAtEnd(){*

*if(head == NULL) printf("\n\t!!!Empty Node list...\n");*

*else {*

*node \*temp = head;*

*while (temp->next != NULL)*

*temp = temp->next;*

*if (temp->prev != NULL)*

*temp->prev->next = NULL;*

*else*

*head = NULL;*

*}*

*}*

*// Function to display the linked list*

*void display() {*

*node \*temp = head;*

*printf("\n\tList of data:\n\t");*

*if (head == NULL) {*

*printf("\t!!! Empty Node list...\n");*

*return;*

*}*

*while (temp != NULL) {*

*printf("%d <-> ", temp->data);*

*temp = temp->next;*

*}*

*printf("NULL\n");*

*}*

*// Menu-driven Dashboard*

*void Dashboard() {*

*int choice, value, pos;*

*do {*

*system("cls");*

*display();*

*printf("\n\tEnter operation: \n");*

*printf("\t\t1. Insert at Beginning\n");*

*printf("\t\t2. Insert at Position\n");*

*printf("\t\t3. Insert at End\n");*

*printf("\t\t4. Delete at Beginning\n");*

*printf("\t\t5. Delete at Position\n");*

*printf("\t\t6. Delete at End\n");*

*printf("\t\t7. Exit\n\n");*

*printf("\tEnter your choice: ");*

*scanf("%d", &choice);*

*switch (choice) {*

*case 1:*

*printf("\n\tEnter value to insert: ");*

*scanf("%d", &value);*

*insertAtBeg(value);*

*break;*

*case 2:*

*printf("\n\tEnter position: ");*

*scanf("%d", &pos);*

*printf("\tEnter value to insert: ");*

*scanf("%d", &value);*

*insertAtPos(pos, value);*

*break;*

*case 3:*

*printf("\n\tEnter value to insert: ");*

*scanf("%d", &value);*

*insertAtEnd(value);*

*break;*

*case 4:*

*deleteAtBeg();*

*break;*

*case 5:*

*printf("\n\tEnter position: ");*

*scanf("%d", &pos);*

*deleteAtPos(pos);*

*break;*

*case 6:*

*deleteAtEnd();*

*break;*

*case 7:*

*printf("\n\tExiting program...\n");*

*break;*

*default:*

*printf("\n\tInvalid choice! Please try again.\n");*

*}*

*printf("\n\tPress Enter to continue...");*

*getch();*

*} while (choice != 7);*

*}*

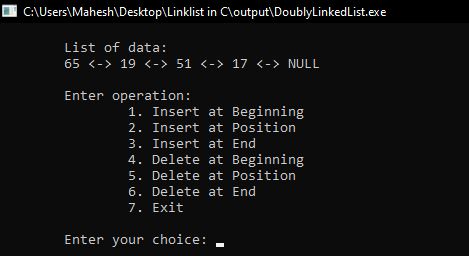
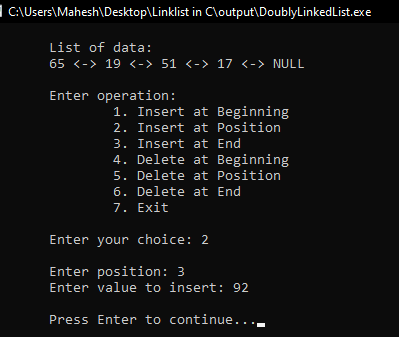
*int main() {*

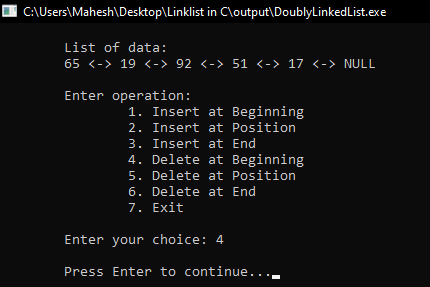
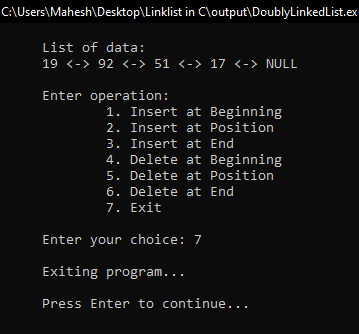
*Dashboard();*

*return 0;*

*}*

Output:

RESULTS AND DISCUSSION:

The stuents are successful to write code for doubly linked list in C programming. This program helps to understand basics of Data structure. The program have menu driven functioality to perform operations in doubly linked list.

CONCLUSION:

This laboratory exercise provided a hands-on experience in DSA. Students gained practical knowledge of implementing basic in doubly linked list and now better equipped to undertake more complex programming tasks in the future.