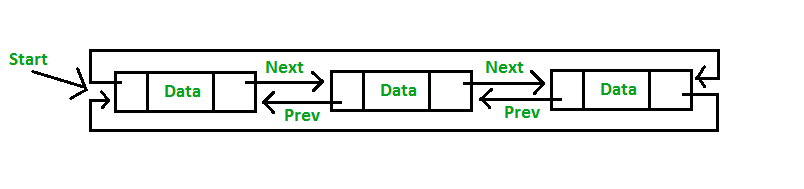
## **LAB 4**

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

**THEORY:**

A "circular doubly linked list" is a type of linked list where each node points to both its previous and next nodes, and the last node in the list connects back to the first node, creating a circular structure that allows for bidirectional traversal through the list.



Operations to perform in circular doubly 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

## PROGRAMS

*#include <stdio.h>*

*#include <stdlib.h>*

*#include <conio.h>*

*// Structure for Circular Doubly Linked List*

*struct Node {*

*int data;*

*struct Node \*next;*

*struct Node \*prev;*

*};*

*typedef struct Node node;*

*node \*head = NULL;*

*// Function to create a new node*

*node\* createNewNode(int item) {*

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

*newnode->data = item;*

*newnode->next = newnode;*

*newnode->prev = newnode;*

*return newnode;*

*}*

*// Function to insert at the beginning*

*void insertAtBeg(int item) {*

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

*if (head == NULL) {*

*head = newnode;*

*} else {*

*node \*tail = head->prev; // Last node*

*newnode->next = head;*

*newnode->prev = tail;*

*tail->next = newnode;*

*head->prev = newnode;*

*head = newnode; // Update head*

*}*

*}*

*// 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 || head == NULL) {*

*insertAtBeg(item);*

*return;*

*}*

*node \*temp = head;*

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

*for (int i = 1; i < pos - 1; i++)*

*temp = temp->next;*

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

*newnode->prev = temp;*

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

*temp->next = newnode;*

*}*

*// Function to insert at the end*

*void insertAtEnd(int item) {*

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

*if (head == NULL) {*

*head = newnode;*

*} else {*

*node \*tail = head->prev; // Last node*

*tail->next = newnode;*

*newnode->prev = tail;*

*newnode->next = head;*

*head->prev = newnode;*

*}*

*}*

*// Function to delete at the beginning*

*void deleteAtBeg() {*

*if (head == NULL) {*

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

*return;*

*}*

*if (head->next == head) { // Only one node*

*free(head);*

*head = NULL;*

*} else {*

*node \*tail = head->prev;*

*node \*delNode = head;*

*head = head->next;*

*head->prev = tail;*

*tail->next = head;*

*free(delNode);*

*}*

*}*

*// Function to delete at a specific position*

*void deleteAtPos(int pos) {*

*if (head == NULL) {*

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

*return;*

*}*

*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 - 1; i++)*

*temp = temp->next;*

*node \*delNode = temp->next;*

*temp->next = delNode->next;*

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

*free(delNode);*

*}*

*// Function to delete at the end*

*void deleteAtEnd() {*

*if (head == NULL) {*

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

*return;*

*}*

*if (head->next == head) { // Only one node*

*free(head);*

*head = NULL;*

*} else {*

*node \*tail = head->prev;*

*node \*newTail = tail->prev;*

*newTail->next = head;*

*head->prev = newTail;*

*free(tail);*

*}*

*}*

*// Function to display the circular doubly linked list*

*void display() {*

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

*if (head == NULL) {*

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

*return;*

*}*

*node \*temp = head;*

*do {*

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

*temp = temp->next;*

*} while (temp != head);*

*printf("(HEAD)\n");*

*}*

*// Menu-driven DSAhboard*

*void DSAhboard() {*

*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);*

*}*

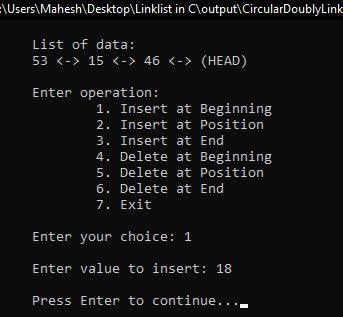
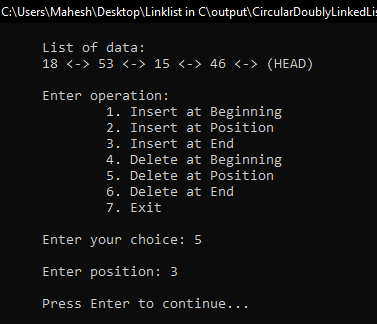
*int main() {*

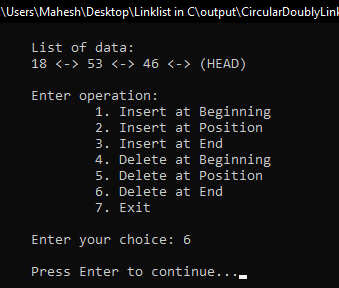
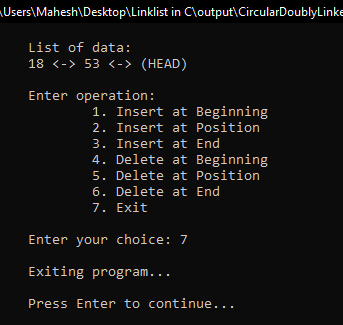
*DSAhboard();*

*return 0;*

*}*

Output:

RESULTS AND DISCUSSION:

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

CONCLUSION:

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