**LAB NO.4**

**EXP.NO:8**

**DATE:30-09-2022**

**STUDENT NAME:VAISHNAVI SINGH**

**ROLL NO:2100290110181**

**AIM / OBJECTIVE: Program for Doubly Linked List Primitive Operations.**

**THEORY:**

**PROEDURE:**

**SOURCE CODE (TYPED):**

**#include <stdio.h>**

**#include <stdlib.h>**

**struct node {**

**int info;**

**struct node \*prev, \*next;**

**};**

**struct node\* start = NULL;**

**void traverse()**

**{**

**if (start == NULL) {**

**printf("\nList is empty\n");**

**return;**

**struct node\* temp;**

**temp = start;**

**while (temp != NULL) {**

**printf("Data = %d\n", temp->info);**

**temp = temp->next;**

**}**

**}**

**}**

**void insertAtFront()**

**{**

**int data;**

**struct node\* temp;**

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

**printf("\nEnter number to be inserted: ");**

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

**temp->info = data;**

**temp->prev = NULL;**

**temp->next = start;**

**start = temp;**

**}**

**void insertAtEnd()**

**{**

**int data;**

**struct node \*temp, \*trav;**

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

**temp->prev = NULL;**

**temp->next = NULL;**

**printf("\nEnter number to be inserted: ");**

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

**temp->info = data;**

**temp->next = NULL;**

**trav = start;**

**if (start == NULL) {**

**start = temp;**

**}**

**else {**

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

**trav = trav->next;**

**temp->prev = trav;**

**trav->next = temp;**

**}**

**}**

**void insertAtPosition()**

**{**

**int data, pos, i = 1;**

**struct node \*temp, \*newnode;**

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

**newnode->next = NULL;**

**newnode->prev = NULL;**

**printf("\nEnter position : ");**

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

**if (start == NULL) {**

**start = newnode;**

**newnode->prev = NULL;**

**newnode->next = NULL;**

**}**

**else if (pos == 1) {**

**insertAtFront();**

**}**

**else {**

**printf("\nEnter number to be inserted: ");**

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

**newnode->info = data;**

**temp = start;**

**while (i < pos - 1) {**

**temp = temp->next;**

**i++;**

**}**

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

**newnode->prev = temp;**

**temp->next = newnode;**

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

**}**

**}**

**void deleteFirst()**

**{**

**struct node\* temp;**

**if (start == NULL)**

**printf("\nList is empty\n");**

**else {**

**temp = start;**

**start = start->next;**

**if (start != NULL)**

**start->prev = NULL;**

**free(temp);**

**}**

**}**

**void deleteEnd()**

**{**

**struct node\* temp;**

**if (start == NULL)**

**printf("\nList is empty\n");**

**temp = start;**

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

**temp = temp->next;**

**if (start->next == NULL)**

**start = NULL;**

**else {**

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

**free(temp);**

**}**

**}**

**void deletePosition()**

**{**

**int pos, i = 1;**

**struct node \*temp, \*position;**

**temp = start;**

**// If DLL is empty**

**if (start == NULL)**

**printf("\nList is empty\n");**

**// Otherwise**

**else {**

**// Position to be deleted**

**printf("\nEnter position : ");**

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

**// If the position is the first node**

**if (pos == 1) {**

**deleteFirst(); // im,proved by Jay Ghughriwala on GeeksforGeeks**

**if (start != NULL) {**

**start->prev = NULL;**

**}**

**free(position);**

**return;**

**}**

**// Traverse till position**

**while (i < pos - 1) {**

**temp = temp->next;**

**i++;**

**}**

**// Change Links**

**position = temp->next;**

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

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

**temp->next = position->next;**

**// Free memory**

**free(position);**

**}**

**}**

**// Driver Code**

**int main()**

**{**

**int choice;**

**while (1) {**

**printf("\n\t1 To see list\n");**

**printf("\t2 For insertion at"**

**" starting\n");**

**printf("\t3 For insertion at"**

**" end\n");**

**printf("\t4 For insertion at "**

**"any position\n");**

**printf("\t5 For deletion of "**

**"first element\n");**

**printf("\t6 For deletion of "**

**"last element\n");**

**printf("\t7 For deletion of "**

**"element at any position\n");**

**printf("\t8 To exit\n");**

**printf("\nEnter Choice :\n");**

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

**switch (choice) {**

**case 1:**

**traverse();**

**break;**

**case 2:**

**insertAtFront();**

**break;**

**case 3:**

**insertAtEnd();**

**break;**

**case 4:**

**insertAtPosition();**

**break;**

**case 5:**

**deleteFirst();**

**break;**

**case 6:**

**deleteEnd();**

**break;**

**case 7:**

**deletePosition();**

**break;**

**case 8:**

**exit(1);**

**break;**

**default:**

**printf("Incorrect Choice. Try Again \n");**

**continue;**

**}**

**}**

**return 0;**

**}**

**OUTPUT:**

