**#5. Linked List**

**Roll Number: CB.EN.P2EBS22007**

**Date of Submission: 25-11-2022**

**Aim:**

To perform following operations in a Linked List using C Programming:

1. Creation of linked list
2. Addition of nodes in the list (start, end and middle)
3. Deletion of nodes in the list (start, end and middle)
4. Print the entire list after each of the above operation

**Tools Required:**

Text editor with C Compiler.

**Experiment:**

Code:

#include <stdio.h>

#include <stdlib.h>

struct node {

int info;

struct node\* link;

};

struct node\* start = NULL;

// Function to create list with n nodes initially

void createList()

{

if (start == NULL) {

int n;

printf("\nEnter the number of nodes: ");

scanf("%d", &n);

if (n != 0) {

int data;

struct node\* newnode;

struct node\* temp;

newnode = malloc(sizeof(struct node));

start = newnode;

temp = start;

printf("\nEnter number to"

" be inserted : ");

scanf("%d", &data);

start->info = data;

for (int i = 2; i <= n; i++) {

newnode = malloc(sizeof(struct node));

temp->link = newnode;

printf("\nEnter number to"

" be inserted : ");

scanf("%d", &data);

newnode->info = data;

temp = temp->link;

}

}

printf("\nThe list is created\n");

}

else

printf("\nThe list is already created\n");

}

void traverse()

{

struct node\* temp;

if (start == NULL)

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

else {

temp = start;

while (temp != NULL) {

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

temp = temp->link;

}

}

}

void insertAtFront()

{

int data;

struct node\* temp;

temp = malloc(sizeof(struct node));

printf("\nEnter number to"

" be inserted : ");

scanf("%d", &data);

temp->info = data;

temp->link = start;

start = temp;

}

// Function to insert at the end of

// the linked list

void insertAtEnd()

{

int data;

struct node \*temp, \*head;

temp = malloc(sizeof(struct node));

printf("\nEnter number to"

" be inserted : ");

scanf("%d", &data);

temp->link = 0;

temp->info = data;

head = start;

while (head->link != NULL) {

head = head->link;

}

head->link = temp;

}

// Function to insert at any specified

// position in the linked list

void insertAtPosition()

{

struct node \*temp, \*newnode;

int pos, data, i = 1;

newnode = malloc(sizeof(struct node));

printf("\nEnter position and data :");

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

temp = start;

newnode->info = data;

newnode->link = 0;

while (i < pos - 1) {

temp = temp->link;

i++;

}

newnode->link = temp->link;

temp->link = newnode;

}

// Function to delete from the front

// of the linked list

void deleteFirst()

{

struct node\* temp;

if (start == NULL)

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

else {

temp = start;

start = start->link;

free(temp);

}

}

// Function to delete from the end

// of the linked list

void deleteEnd()

{

struct node \*temp, \*prevnode;

if (start == NULL)

printf("\nList is Empty\n");

else {

temp = start;

while (temp->link != 0) {

prevnode = temp;

temp = temp->link;

}

free(temp);

prevnode->link = 0;

}

}

// Function to delete from any specified

// position from the linked list

void deletePosition()

{

struct node \*temp, \*position;

int i = 1, pos;

if (start == NULL)

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

else {

printf("\nEnter index : ");

scanf("%d", &pos);

position = malloc(sizeof(struct node));

temp = start;

while (i < pos - 1) {

temp = temp->link;

i++;

}

position = temp->link;

temp->link = position->link;

free(position);

}

}

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("\t0 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 0:

exit(1);

break;

default:

printf("Incorrect Choice\n");

}

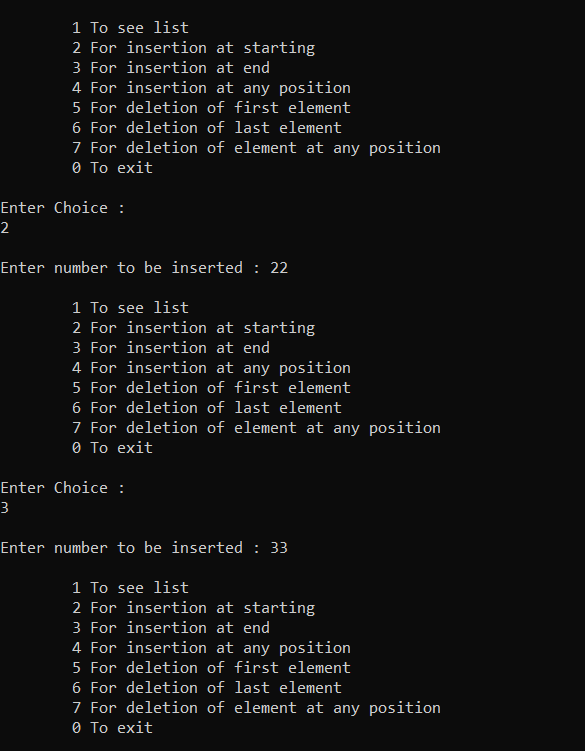
}

return 0;

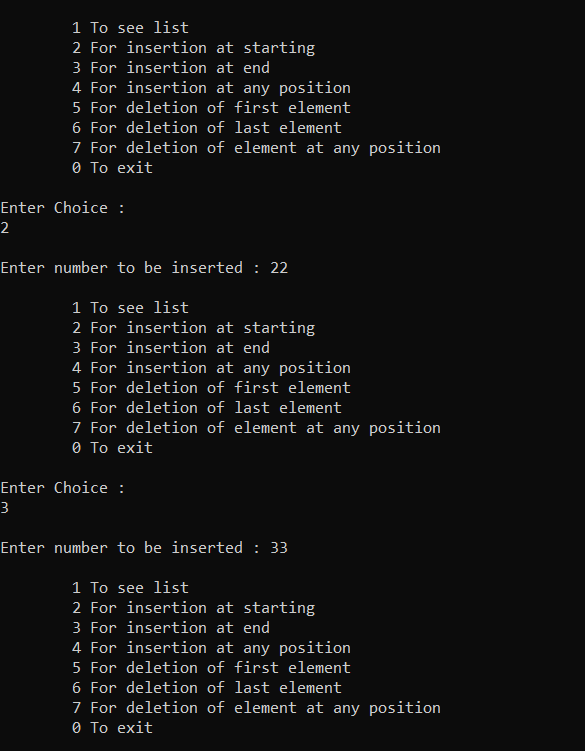
}

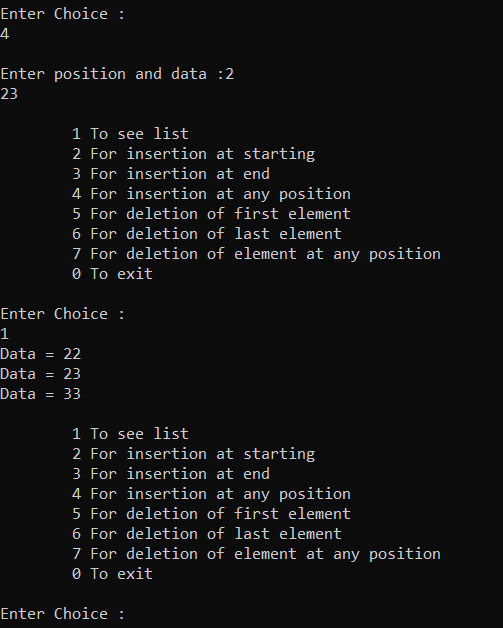
Result

1. Creation of linked list

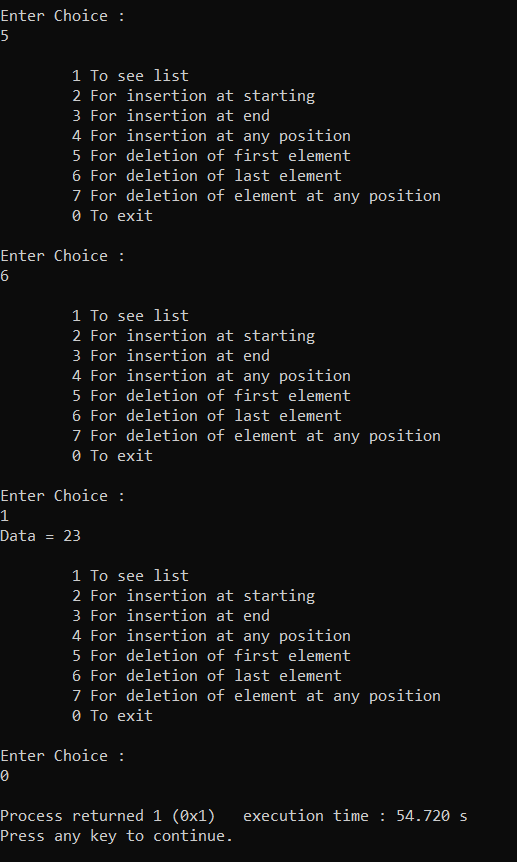


1. Addition of nodes in the list (start, end and middle)

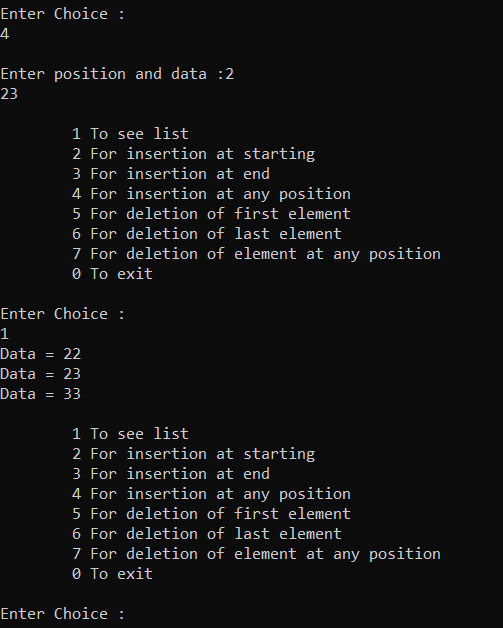


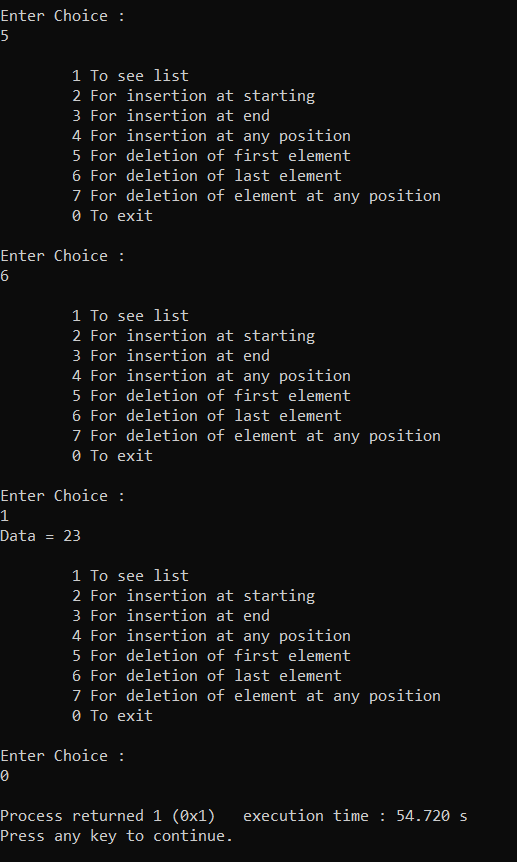


1. Deletion of nodes in the list (start, end and middle)



1. Print the entire list after each of the above operation





**Inference and Result:**

Singly Linked List is created using C program and all the operations like addition, deletion and printing the list is executed properly.