**#5. Linked List**

**Roll Number:**

**Date of Submission:**

**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:**

// Online C compiler to run C program online

#include <stdio.h>

#include<stdlib.h>

typedef struct node{

int data;

struct node\* next;

}N;

N \*head=NULL,\*newNode,\*temp;

N\* nodeCreate(){

int choice=1;

while(choice){

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

printf("Enter the data:");

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

if(head==NULL){

head=newNode;

temp=newNode;

}

else{

temp->next=newNode;

temp=newNode;

}

printf("DO YOU WISH TO ENTER THE ELEMENTS(1 or 0):");

scanf("%d",&choice);

while(choice!=1&&choice!=0){

printf("Enter 1 or 0\n");

printf("DO YOU WISH TO ENTER THE ELEMENTS(1 or 0):");

scanf("%d",&choice);

}

}

}

nodeDisplay(){

temp=head;

printf("The elements in list are:");

while(temp!=0){

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

if(temp->next!=NULL)

printf("->");

temp=temp->next;

}

printf("\n");

}

void insertAtbegining(){

temp=head;

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

printf("\nEnter the element needs to be enetered at begin:");

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

newNode->next=temp;

head=newNode;

}

void insertAtEnd(){

temp=head;

while(temp->next!=0){

temp=temp->next;

}

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

printf("\nEnter the element which needs to be enetered at the end:");

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

newNode->next=0;

temp->next=newNode;

}

void insertAtMiddle(){

int ch,count,i=0;

count=countOfElements();

printf("\nThere are %d elements in the linked list where do you want to enter a new element:",count);

scanf("%d",&ch);

if(count>=ch){

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

temp=head;

while(i<ch){

temp=temp->next;

i++;

}

printf("Enter a element which needs to entered at %d the position:",ch);

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

newNode->next=temp->next;

temp->next=newNode;

}

else{

printf("Enter a valid number");

}

}

int countOfElements(){

temp=head;

int count=0;

while(temp!=0){

temp=temp->next;

count++;

}

return count;

}

void deleteAtbegining(){

temp=head;

head=head->next;

free(temp);

}

void deleteAtEnd(){

N\* prevNode;

temp=head;

while(temp->next!=0){

prevNode=temp;

temp=temp->next;

}

if(temp==head){

head=NULL;

}

else

prevNode->next=NULL;

free(temp);

}

void deleteAtRandom(){

int ch,i=0,count;

N\* prevNode;

printf("where do you want to remove an element:");

scanf("%d",&ch);

count=countOfElements();

if(count>=ch){

temp=head;

while(i<ch){

prevNode=temp;

temp=temp->next;

i++;

}

prevNode->next=temp->next;

free(temp);

}

else{

printf("Enter a valid number");

}

}

int main() {

int choice;

printf("Create a Node\n");

nodeCreate();

nodeDisplay();

while(1){

printf("\n1-Insert Node at beginning\n2-Insert Node at End\n3-Insert Node at middle\n4-Delete at beginning\n5-Delete at end\n6-Delete Node at Middle\n7-Exit\nEnter your choice:");

scanf("%d",&choice);

switch(choice){

case 1: insertAtbegining();

nodeDisplay();

break;

case 2:insertAtEnd();

nodeDisplay();

break;

case 3: insertAtMiddle();

nodeDisplay();

break;

case 4: deleteAtbegining();

nodeDisplay();

break;

case 5: deleteAtEnd();

nodeDisplay();

break;

case 6: deleteAtRandom();

nodeDisplay();

break;

case 7: exit;

break;

default: printf("Enter a valid number");

return 0;

}

if(choice==7)

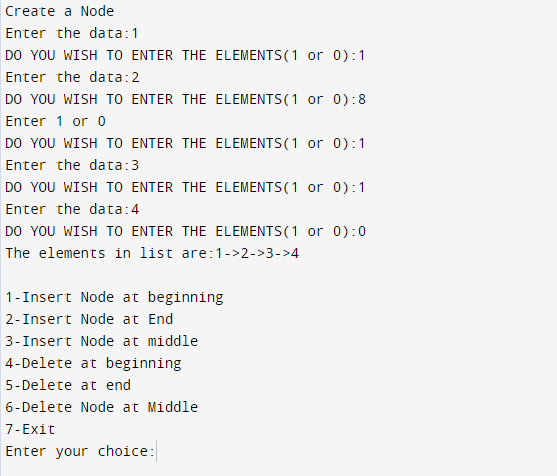
break;

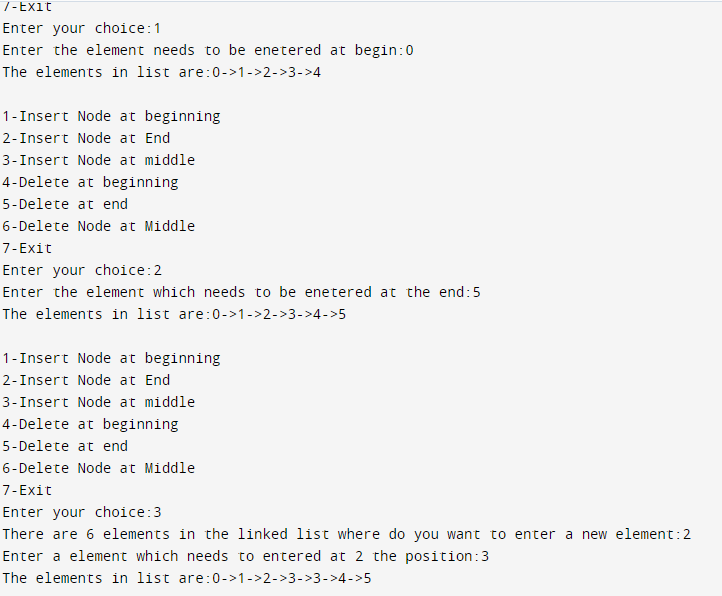
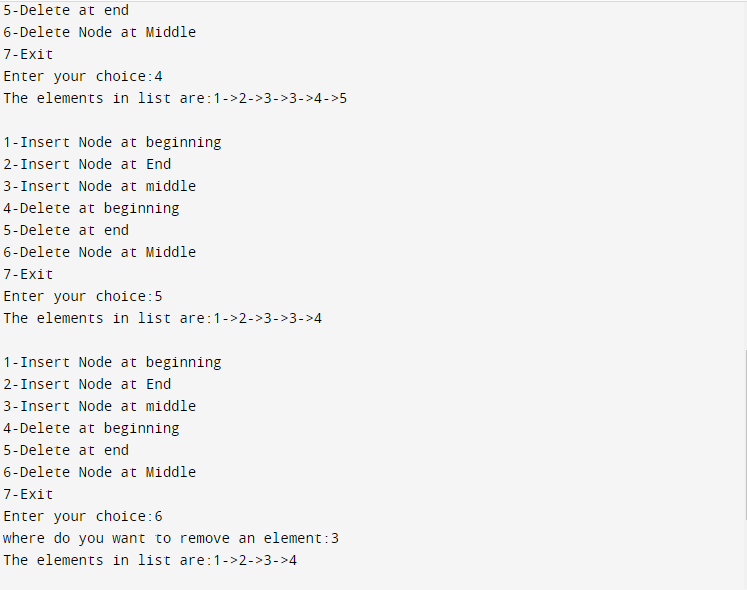
}

}

**Result:**

1. **Creation of linked list**



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

**Inference and Result:**

Thus, singly linked list has been implemented and various operations like insertion at beginning, end, middle has been implemented and same has been implemented for deletion also.