**Linked list : Insertion and Deletion(at beginning, at end, at a given position)**

PROGRAM:

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct node{

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node\* createnode(int data) {

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

newnode->data = data;

newnode->next = NULL;

return newnode;

}

void insert\_at\_beg(int data) {

struct node\* newnode = createnode(data);

newnode->next = head;

head = newnode;

}

void insert\_at\_end(int data) {

struct node\* newnode = createnode(data);

if (head == NULL) {

head = newnode;

return;

}

struct node\* temp = head;

while (temp->next != NULL) {

temp = temp->next;

}

temp->next = newnode;

}

void insert\_at\_pos(int data, int pos) {

struct node\* newnode = createnode(data);

if (pos == 1) {

newnode->next = head;

head = newnode;

return;

}

struct node\* temp = head;

for (int i = 0; i < pos - 2 && temp != NULL; i++) {

temp = temp->next;

}

if (temp == NULL) {

printf("Invalid position\n");

free(newnode);

return;

}

newnode->next = temp->next;

temp->next = newnode;

}

void delete\_at\_beg() {

if (head == NULL) {

printf("List is empty\n");

return;

}

struct node\* temp = head;

head = head->next;

free(temp);

}

void delete\_at\_end() {

if (head == NULL) {

printf("List is empty\n");

return;

}

struct node\* temp = head;

if (temp->next == NULL) {

head = NULL;

free(temp);

return;

}

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

temp = temp->next;

}

free(temp->next);

temp->next = NULL;

}

void delete\_at\_pos(int pos) {

if (head == NULL) {

printf("List is empty\n");

return;

}

if (pos == 1) {

struct node\* temp = head;

head = head->next;

free(temp);

return;

}

struct node\* temp = head;

for (int i = 0; i < pos - 2 && temp != NULL; i++) {

temp = temp->next;

}

if (temp == NULL || temp->next == NULL) {

printf("Position not valid\n");

return;

}

struct node\* deletenode = temp->next;

temp->next = deletenode->next;

free(deletenode);

}

void display() {

if (head == NULL) {

printf("List is empty\n");

return;

}

struct node \*temp = head;

while (temp != NULL) {

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

temp = temp->next;

}

printf("NULL\n");

}

int main() {

int option, data, position;

while (1) {

printf("\n1. Insert at beginning\n2. Insert at end\n3. Insert at position\n4. Delete at beginning\n5. Delete at end\n6. Delete at position\n7. Display\n8. Exit\n");

printf("Enter your choice: ");

scanf("%d", &option);

switch (option) {

case 1:

printf("Enter data to insert at beginning: ");

scanf("%d", &data);

insert\_at\_beg(data);

break;

case 2:

printf("Enter data to insert at end: ");

scanf("%d", &data);

insert\_at\_end(data);

break;

case 3:

printf("Enter data and position to insert: ");

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

insert\_at\_pos(data, position);

break;

case 4:

delete\_at\_beg();

break;

case 5:

delete\_at\_end();

break;

case 6:

printf("Enter position to delete: ");

scanf("%d", &position);

delete\_at\_pos(position);

break;

case 7:

display();

break;

case 8:

return 0; // Exit gracefully

default:

printf("Invalid choice\n");

}

}

return 0;

}

OUTPUT:









