**Single Linked List**

#include <stdio.h>

#include <stdlib.h>

struct Node {

int data;

struct Node \*next;

};

struct Node \*head = NULL;

// Insert at end

void insert(int val) {

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

newNode->data = val;

newNode->next = NULL;

if(head == NULL) {

head = newNode;

} else {

struct Node \*temp = head;

while(temp->next != NULL)

temp = temp->next;

temp->next = newNode;

}

printf("%d inserted.\n", val);

}

// Delete from end

void deleteEnd() {

if(head == NULL) {

printf("List is empty.\n");

return;

}

if(head->next == NULL) {

printf("Deleted: %d\n", head->data);

free(head);

head = NULL;

return;

}

struct Node \*temp = head;

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

temp = temp->next;

printf("Deleted: %d\n", temp->next->data);

free(temp->next);

temp->next = NULL;

}

// Display list

void display() {

if(head == NULL) {

printf("List is empty.\n");

return;

}

struct Node \*temp = head;

printf("List elements: ");

while(temp != NULL) {

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

temp = temp->next;

}

printf("\n");

}

int main() {

int choice, val;

while(1) {

printf("\n1. Insert 2. Delete 3. Display 4. Exit\nEnter choice: ");

scanf("%d", &choice);

switch(choice) {

case 1:

printf("Enter value to insert: ");

scanf("%d", &val);

insert(val);

break;

case 2:

deleteEnd();

break;

case 3:

display();

break;

case 4:

exit(0);

default:

printf("Invalid choice.\n");

}

}

return 0;

}

**Output**

