1. **Insert numbers in a list**

**Code:**

#include <stdio.h>

#include <stdlib.h>

// Define a structure for the node of the list

struct Node {

int data;

struct Node\* next;

};

// Function to create a new node

struct Node\* createNode(int data) {

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

if (newNode == NULL) {

printf("Memory allocation failed.\n");

exit(1);

}

newNode->data = data;

newNode->next = NULL;

return newNode;

}

// Function to insert a node at the end of the list

void insertNode(struct Node\*\* head, int data) {

if (\*head == NULL) {

\*head = createNode(data);

} else {

struct Node\* temp = \*head;

while (temp->next != NULL) {

temp = temp->next;

}

temp->next = createNode(data);

}

}

// Function to display the list

void displayList(struct Node\* head) {

printf("List: ");

while (head != NULL) {

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

head = head->next;

}

printf("\n");

}

int main() {

int num;

char choice;

struct Node\* head = NULL; // Initialize an empty list

// Insert initial numbers into the list

printf("Enter initial numbers into the list (enter -1 to stop):\n");

while (1) {

scanf("%d", &num);

if (num == -1)

break;

insertNode(&head, num);

}

// Display the initial list

displayList(head);

// Insert a number into the list

printf("Enter a number to insert: ");

scanf("%d", &num);

insertNode(&head, num);

// Display the updated list

printf("Updated ");

displayList(head);

// Free dynamically allocated memory to prevent memory leaks

struct Node\* temp;

while (head != NULL) {

temp = head;

head = head->next;

free(temp);

}

return 0;

}

**Output:**

Enter initial numbers into the list (enter -1 to stop):

5

4 12 78 45 03

-1

List: 5 4 12 78 45 3

Enter a number to insert: 13

Updated List: 5 4 12 78 45 3 13

--------------------------------

Process exited after 26.85 seconds with return value 0

Press any key to continue . . .

