**1.INSERTION IN SINGLY LINKED LIST:**

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

}\*head;

void createList(int n);

void insertNodeAtBeginning(int data);

void displayList();

int main()

{

int n, data;

printf("Enter the total number of nodes: ");

scanf("%d", &n);

createList(n);

printf("\nData in the list \n");

displayList();

printf("\nEnter data to insert at beginning of the list: ");

scanf("%d", &data);

insertNodeAtBeginning(data);

printf("\nData in the list \n");

displayList();

return 0;

}

void createList(int n)

{

struct node \*newNode, \*temp;

int data, i;

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

if(head == NULL)

{

printf("Unable to allocate memory.");

}

else

{

printf("Enter the data of node 1: ");

scanf("%d", &data);

head->data = data;

head->next = NULL;

temp = head;

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

{

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

if(newNode == NULL)

{

printf("Unable to allocate memory.");

break;

}

else

{

printf("Enter the data of node %d: ", i);

scanf("%d", &data);

newNode->data = data;

newNode->next = NULL;

temp->next = newNode;

temp = temp->next;

}

}

printf("SINGLY LINKED LIST CREATED SUCCESSFULLY\n");

}

}

void insertNodeAtBeginning(int data)

{

struct node \*newNode;

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

if(newNode == NULL)

{

printf("Unable to allocate memory.");

}

else

{

newNode->data = data;

newNode->next = head;

head = newNode;

printf("DATA INSERTED SUCCESSFULLY\n");

}

}

void displayList()

{

struct node \*temp;

if(head == NULL)

{

printf("List is empty.");

}

else

{

temp = head;

while(temp != NULL)

{

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

temp = temp->next;

}

}

}

**OUTPUT:**

Enter the total number of nodes: 4

Enter the data of node 1: 8

Enter the data of node 2: 9

Enter the data of node 3: 10

Enter the data of node 4: 11

SINGLY LINKED LIST CREATED SUCCESSFULLY

Data in the list

Data = 8

Data = 9

Data = 10

Data = 11

Enter data to insert at beginning of the list: 1

DATA INSERTED SUCCESSFULLY

Data in the list

Data = 1

Data = 8

Data = 9

Data = 10

Data = 11

**2.DELETION IN SINGLY LINKED LIST:**

#include <stdio.h>

#include <stdlib.h>

struct node {

int x;

struct node \*next;

};

struct node \*head;

struct node \*create\_item(int x);

void print\_list();

void delete\_item(int x);

int main(int argc, int argv) {

struct node \*n;

int i;

head = create\_item(1);

n = head;

for (i = 2; i < 10; i++) {

n->next = create\_item(i);

n = n->next;

}

print\_list();

delete\_item(7);

print\_list();

delete\_item(1);

delete\_item(3);

delete\_item(5);

delete\_item(9);

print\_list();

}

struct node \*create\_item(int x) {

struct node \*new;

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

new->x = x;

return new;

}

void print\_list() {

struct node \*iter;

iter = head;

while (iter != NULL) {

printf("num: %i\n", iter->x);

iter = iter->next;

}

}

void delete\_item(int x) {

struct node \*iter;

iter = head;

if (iter == NULL) {

printf("not found\n");

return;

}

if (iter->x == x) {

printf("found in first element: %i\n", x);

head = head->next;

return;

}

while (iter->next != NULL) {

if (iter->next->x == x) {

printf("deleting element: %i\n", x);

iter->next = iter->next->next;

return;

}

iter = iter->next;

}

printf("not found\n");

}

**OUTPUT:**

num: 1

num: 2

num: 3

num: 4

num: 5

num: 6

num: 7

num: 8

num: 9

deleting element: 7

num: 1

num: 2

num: 3

num: 4

num: 5

num: 6

num: 8

num: 9

found in first element: 1

deleting element: 3

deleting element: 5

deleting element: 9

num: 2

num: 4

num: 6

num: 8