**Linked list implementation**

#include <stdio.h>

#include <stdlib.h>

struct Node{

int data;

struct Node \*next;

};

typedef struct Node NodeType;

NodeType \*head = NULL;

NodeType \*getnode(){

NodeType \*p;

p=(NodeType\*)malloc(sizeof(NodeType));

return(p);

}

void display(){

// printf("linked list elements:\n");

if (head == NULL){

printf("Linked list is empty\n");

}

else{

NodeType \*ptr;

ptr = head;

printf("[ ");

while (ptr != NULL){

if (ptr->next == NULL){

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

}else{

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

}

ptr = ptr->next;

}

printf(" ]\n");

}

}

void insertEnd(int x){

NodeType \*p;

p = (NodeType\*)malloc(sizeof(NodeType));

p->data = x;

p->next = NULL;

if (head == NULL){

head = p;

}else{

NodeType \*ptr = head;

while (ptr->next!=NULL){

ptr = ptr->next;

}

ptr->next = p;

}

}

void insertBeg(int x){

NodeType \*ptr = getnode();

ptr->data = x;

ptr->next = head;

head = ptr;

}

void insert(int x,int index){

if (head == NULL){

printf("Empty List: index out of range");

return;

}

if (index ==0){

insertBeg(x);

return;

}

NodeType \*prev = getnode();

NodeType \*ptr = head;

// ptr = head;

for (int i=0;i<index;i++){

prev = ptr;

ptr = ptr->next;

}

NodeType \*p =getnode();

p->data = x;

p->next = ptr;

prev->next = p;

}

void deleteEnd(){

if (head==NULL){

printf("Cannot delete. List is already empty\n");

return;

}

NodeType \*prev = getnode();

NodeType \*ptr = head;

while (ptr->next!=NULL){

prev = ptr;

ptr = ptr->next;

}

prev->next = NULL;

free(ptr);

}

void deleteBeg(){

if (head==NULL){

printf("Cannot delete. List is already empty\n");

return;

}

NodeType \*ptr = head;

head = ptr->next;

free(ptr);

}

void del(int index){

if (head==NULL){

printf("Cannot delete. List is already empty\n");

return;

}

if (index == 0){

deleteBeg;

return;

}

NodeType \*ptr = head;

NodeType \*prev;

for (int i=0;i<index;i++){

prev = ptr;

ptr = ptr->next;

}

prev->next = ptr->next;

free(ptr);

}

int main(){

insertEnd(1); // add 1

display();

insertEnd(2); // add 2

display();

insertEnd(3); // add 3

display();

insertBeg(5); // add 5 at beginning

display();

insert(4,1); // add 4 at 1st index

display();

deleteEnd(); // delete end elemnt

display();

deleteBeg(); // delete 1st element

display();

del(1); //delete elem at index 1

display();

}

