```c
// Author: Gokul Raj, 235, R3A, CSE
// QUESTION: Implement a stack using Linked List.
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// Structure to hold Singly Linked List Nodes for a Stack.
struct Node {
    int data;
    struct Node *prev;
};

// TOP of Stack
struct Node *stack = NULL;

// Function to check if stack is empty
int isEmpty(){
    // returns 1 if empty, else 0
    if(stack == NULL)
        return 1;
    return 0;
}

// Push new element onto stack
void push(int ele){
    struct Node *newNode = malloc(sizeof(struct Node));
    newNode->prev = stack;
    newNode->data = ele;
    stack = newNode;
}

// Pops out top element of stack
int pop(){
    struct Node *temp = stack;
    stack = stack->prev;
    int op = temp->data;
    free(temp);
    return op;
}
```

```c
// Displays stack contents without affecting it.
void showStack(){
    if(isEmpty()){
        printf("Stack Empty!!\n");
        return;
    }
    char *op = malloc(1024);
    char *tmp = malloc(1024);
    op[0] = '\0';
    for(struct Node *p = stack; p != NULL; p = p->prev){
        sprintf(tmp, "%d | ", p->data);
        strcat(op, tmp);
    }
    int len = strlen(op);
    len = len-2;
    op[len] = '\0';
    for(int i=0; i<len; i++)
        tmp[i] = '\xCD';
    tmp[len] = '\0';
    printf("       %s\xBB\nTOP => %s\xBA\n       %s\xBC\n", tmp, op, tmp);
    free(op);
    free(tmp);
}

// Driver Code for UI
void main(){
    int choice, ele, looping = 1;
    printf("Linked List Stack\nChoose an Operation:\n[1] Push\t[2] Pop\n[3] Show Stack\t[4]
EXIT\n");
    while(looping){
        printf("Choice: ");
        scanf("%d", &choice);
        switch(choice){
            case 1:
                printf("Element to push: ");
                scanf("%d", &ele);
                push(ele);
                break;
            case 2:
                if(isEmpty()){
                    printf("Stack Empty!!\n");
                    break;
                }
                ele = pop();
                printf("GOT: %d\n", ele);
                break;
            case 3:
                showStack();
                break;
            case 4:
                looping = 0;
                break;
            default:
                printf("Invalid Choice.. Try Again.\n");
        }
    }
}
```

OUTPUT

```
Linked List Stack
Choose an Operation:
[1] Push        [2] Pop
[3] Show Stack  [4] EXIT
Choice: 3
Stack Empty!!
Choice: 1
Element to push: 100
Choice: 1
Element to push: 200
Choice: 1
Element to push: 300
Choice: 1
Element to push: 400
Choice: 3
╔═══════════════════════════╗
TOP => 400 | 300 | 200 | 100
╚═══════════════════════════╝
Choice: 2
GOT: 400
Choice: 3
╔═══════════════╗
TOP => 300 | 200 | 100
╚═══════════════╝
Choice: 2
GOT: 300
Choice: 3
╔═══════╗
TOP => 200 | 100
╚═══════╝
Choice: 2
GOT: 200
Choice: 3
╔═══╗
TOP => 100
╚═══╝
Choice: 1
Element to push: 50
Choice: 3
╔═══════╗
TOP => 50 | 100
╚═══════╝
Choice: 4
```