//. Write a program to implement stack using linked list. Perform necessary operations.

#include <stdio.h>

#include <stdlib.h>

struct node

{

    int data;

    struct node \*next;

};

struct node \*top = NULL;

int isEmpty()

{

    return top == NULL;

}

void push(int data)

{

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

    new\_node->data = data;

    new\_node->next = top;

    top = new\_node;

}

int pop()

{

    if (top == NULL)

    {

        printf("Stack is empty\n");

        return -1;

    }

    int data = top->data;

    struct node \*temp = top;

    top = top->next;

    free(temp);

    return data;

}

int main()

{

    push(10);

    push(20);

    push(30);

    printf("Popped element: %d\n", pop());

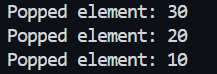
    printf("Popped element: %d\n", pop());

    printf("Popped element: %d\n", pop());

    return 0;

}

Output:



// Write a program to input an n digit number. Now break this number into its individual

// digits and then store every single digit in a separate node thereby forming a linked list.For

//  Example,

// you entered 235, then there will be 3 nodes in the list containing nodes with value 2, 3, 5.

#include <stdio.h>

#include <stdlib.h>

struct node

{

    int data;

    struct node \*next;

};

struct node \*head = NULL;

int main()

{

    int n;

    printf("Enter a number: ");

    scanf("%d", &n);

    int rem;

    while (n > 0)

    {

        rem = n % 10;

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

        new\_node->data = rem;

        new\_node->next = head;

        head = new\_node;

        n /= 10;

    }

    struct node \*temp = head;

    while (temp != NULL)

    {

        printf("%d ", temp->data);

        temp = temp->next;

    }

    return 0;

}



//// Q2:Write a program to reverse the element of ABOVE singly linked list.

#include <stdio.h>

#include <stdlib.h>

struct node

{

    int data;

    struct node \*next;

};

struct node \*head = NULL;

// Function to reverse the linked list

struct node \*reverse(struct node \*head)

{

    struct node \*prev = NULL, \*curr = head, \*next = NULL;

    while (curr != NULL)

    {

        next = curr->next; // Store next node

        curr->next = prev; // Reverse current node's pointer

        prev = curr;       // Move prev and curr one step forward

        curr = next;

    }

    return prev; // New head of the reversed list

}

// Function to print the linked list

void printList(struct node \*head)

{

    struct node \*temp = head;

    while (temp != NULL)

    {

        printf("%d ", temp->data);

        temp = temp->next;

    }

    printf("\n");

}

int main()

{

    int n;

    printf("Enter a number: ");

    scanf("%d", &n);

    int rem;

    // Creating linked list from digits of the number

    while (n > 0)

    {

        rem = n % 10;

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

        new\_node->data = rem;

        new\_node->next = head;

        head = new\_node;

        n /= 10;

    }

    printf("Original Linked List: ");

    printList(head);

    // Reverse the linked list

    head = reverse(head);

    printf("Reversed Linked List: ");

    printList(head);

    return 0;

}

