```
//AL2 - Aula de 12/06/2017
//Pilhas din?micas/encadeadas

#include <stdio.h>
#include <stdlib.h>

//Defini??o de constantes
#define FALSE 0
#define TRUE 1

//Defini??o de tipos

//definindo o struct que representar? cada posi??o da lista, denominado
de N?.
//Cada N? conter? o elemento da lista e o ponteiro para o pr?ximo
elemento
typedef struct No {
        int valor;                      //cada valor sendo manipulado
pela lista
        struct No* prox;        //endere?o do n? que cont?m o pr?ximo
elemento
} TNo;

typedef TNo* TLista;

//prot?tipos das fun??es
int empilhar (TLista* L, int numero);
int desempilhar (TLista* L, int *numero);
int topo (TLista* L, int *numero);
void destruir (TLista* L);

int menu ();

//MAIN
int main ()
{
        TLista L = NULL;   //criando a lista, inicialmente vazia
        int op, num;

        do
        {
                system ("cls");
                op = menu ();

                switch (op)
                {
                        case 1: //Empilhar
                                        printf ("Entre com o elemento a
ser empilhado: ");
                                        scanf ("%d", &num);

                                        if (empilhar (&L, num) == TRUE)
                                        {
                                                printf ("Elemento %d
empilhado com sucesso!\n", num);
                                        }
                                        else
```

```
                                        {
                                                printf ("ERRO: Elemento
%d nao empilhado!\n", num);
                                        }
                                        break;

                        case 2: //Desempilhar

                                        if (desempilhar (&L, &num) ==
TRUE)
                                        {
                                                printf ("Elemento %d foi
desempilhado!\n", num);
                                        }
                                        else
                                        {
                                                printf ("ERRO: pilha
vazia!\n");
                                        }
                                        break;

                        case 3: //Topo
                                        if (topo (&L, &num) == TRUE)
                                        {
                                                printf ("Topo da pilha:
%d!\n", num);
                                        }
                                        else
                                        {
                                                printf ("ERRO: pilha
vazia!\n");
                                        }
                                        break;

                        case 4: //Destruir
                                        destruir (&L);
                                        break;

                        case 5: //Sair
                                        printf ("Fim do programa!\n");
                }
                system ("pause");
        }
        while (op != 5);
}

//Implementa??es

//Insere um elemento no topo da pilha, retornando TRUE ou FALSE
int empilhar (TLista* L, int numero)
{
        TLista aux = (TLista) malloc (sizeof(TNo));

        if (!aux)
        {
                return FALSE;   //sem mem?ria dispon?vel
        }
```

```
        else
        {
                aux->valor = numero;

                aux->prox = *L;

                *L = aux;

                return TRUE;
        }
}

//Remove o elemento que encontra-se no topo da pilha
//(e o retorna, caso exista), retornando TRUE ou FALSE
int desempilhar (TLista* L, int *numero)
{
        TLista aux;

        if (*L != NULL)   //testando se a pilha n?o est? vazia
        {
                aux = *L;
                *L = (*L)->prox;

                *numero = aux->valor;

                free (aux);

                return TRUE;
        }
        else
        {
                return FALSE;
        }
}

//Retorna em numero o elemento que encontra-se no topo da pilha,
retornando TRUE
//ou FALSE
int topo (TLista* L, int *numero)
{
        if (*L != NULL)   //testando se a pilha n?o est? vazia
        {
                *numero = (*L)->valor;

                return TRUE;
        }
        else
        {
                return FALSE;
        }
}


//Destr?i todos os elementos da Lista (vers?o iterativa)
void destruir (TLista* L)
{
        TLista aux;
```

```
        while (*L)
        {
                aux = *L;
                *L = (*L)->prox;   //ou *L = aux->prox;
                free (aux);
        }
}

//Exibe um menu ao usu?rio, retornando a op??o escolhida
int menu ()
{
        int opcao;

        printf ("Menu de opcoes:\n");
        printf ("(1) Empilhar\n(2) Desempilhar\n(3) Topo\n(4)
Destruir\n(5) Sair\n\n");
        printf ("Entre com a sua opcao: ");
        scanf ("%d", &opcao);

        return opcao;
}
```