



# INSTITUTO FEDERAL

## Ceará

Nome: Ana Vitória Cabral Duarte  
Matrícula: 20212013020205

## Avaliação-09

Implementar uma Pilha usando uma lista Encadeada usando as seguintes operações:

- 1) Verificar o Topo da Pilha. Consiste na verificação da existência de um elemento na Pilha;
- 2) Implementar o Push (é a inserção de um elemento na Pilha alterando a informação do Topo);
- 3) Implementar o Pop (é a retirada de um elemento da Pilha, ou seja é alterado o Topo com o elemento logo abaixo).
- 4) Evidenciar a execução das operações acima mostrando o conteúdo da pilha a partir do Topo.

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

```
// Definição de um nó da pilha
typedef struct Node {
    int data;
    struct Node* next;
} Node;
```

```
// Estrutura da pilha
typedef struct Stack {
    Node* top;
} Stack;
```

```
// Função para criar uma nova pilha
Stack* createStack() {
    Stack* stack = (Stack*)malloc(sizeof(Stack));
    stack->top = NULL;
    return stack;
}
```

```
// Função para verificar se a pilha está vazia
int isEmpty(Stack* stack) {
    setlocale(LC_ALL, "portuguese");
```

```

    return stack->top == NULL;
}

// Função para empurrar (push) um elemento para a pilha
void push(Stack* stack, int data) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->data = data;
    newNode->next = stack->top;
    stack->top = newNode;
    printf("Elemento %d empilhado\n", data);
}

// Função para desempilhar (pop) um elemento da pilha
int pop(Stack* stack) {
    if (isEmpty(stack)) {
        printf("A pilha está vazia. Não é possível desempilhar.\n");
        return -1;
    }
    Node* topNode = stack->top;
    int data = topNode->data;
    stack->top = topNode->next;
    free(topNode);
    return data;
}

// Função para mostrar o conteúdo da pilha a partir do topo
void display(Stack* stack) {
    if (isEmpty(stack)) {
        printf("A pilha está vazia.\n");
        return;
    }

    Node* current = stack->top;
    printf("Conteúdo da pilha a partir do topo: ");
    while (current != NULL) {
        printf("%d ", current->data);
        current = current->next;
    }
    printf("\n");
}

int main() {
    Stack* stack = createStack();

    display(stack);

    push(stack, 10);
    push(stack, 20);
    push(stack, 30);

    display(stack);
}

```

```

int topValue = stack->top->data;
printf("O topo da pilha contém: %d\n", topValue);

int poppedValue = pop(stack);
if (poppedValue != -1) {
    printf("Elemento %d desempilhado\n", poppedValue);
}

display(stack);

return 0;
}

```

The screenshot displays the Dev-C++ IDE with a C++ program for a stack. The code is as follows:

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <locale.h>
4
5
6 // Definição de um nó da pilha
7 typedef struct Node {
8     int data;
9     struct Node* next;
10 } Node;
11
12 // Estrutura da pilha
13 typedef struct Stack {
14     Node* top;
15 } Stack;
16
17 // Função para criar uma nova pilha
18 Stack* createStack() {
19     Stack* stack = (Stack*)malloc(sizeof(Stack));
20     stack->top = NULL;
21     return stack;
22 }
23
24 // Função para verificar se a pilha está vazia
25 int isEmpty(Stack* stack) {
26     setlocale(LC_ALL, "portuguese");
27
28     return stack->top == NULL;
29 }
30

```

The output window shows the following results:

```

A pilha está vazia.
Elemento 10 empilhado
Elemento 20 empilhado
Elemento 30 empilhado
Conteúdo da pilha a partir do topo: 30 20 10
O topo da pilha contém: 30
Elemento 30 desempilhado
Conteúdo da pilha a partir do topo: 20 10

-----
Process exited after 1.63 seconds with return value 0
Pressione qualquer tecla para continuar. . .

```

The compiler log at the bottom shows the following output:

```

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Alunos\Documents\Windows\Untitled3.exe
- Output Size: 130,625 KiB
- Compilation Time: 0,20s

```

The status bar at the bottom indicates the current line and column: Line: 26, Col: 33.