

**ASSOCIAÇÃO EDUCACIONAL DE VITÓRIA
FACULDADES INTEGRADAS SÃO PEDRO
CURSO DE GRADUAÇÃO EM TECNOLOGIA EM ANÁLISE E
DESENVOLVIMENTO DADOS**

ENRICO DA LAQUA DOS REIS

ESTRUTURA DE DADOS – PILHAS

**VITÓRIA
2021**

ENRICO DA LAQUA DOS REIS

ESTRUTURA DE DADOS – PILHAS

Trabalho acadêmico do Curso de Graduação em Tecnologia em Análise e Desenvolvimento de Dados, apresentado às Faculdades Integradas São Pedro como parte das exigências da disciplina Estrutura de Dados, sob orientação do(a) professor(a) Jarbas Ferreira da Silva Araújo.

VITÓRIA

2021

SUMÁRIO

1 INTRODUÇÃO.....	1
2 CÓDIGO PROPOSTO	2
3 CORREÇÃO DO CÓDIGO	3
4 COMPILAÇÃO OBTIDA.....	7
5 CONCLUSÃO	10
REFERÊNCIAS	11

1 INTRODUÇÃO

As *Pilhas* são estruturas de dados do tipo Last In, First Out (LIFO), ou seja, o último elemento a ser inserido na estrutura, será o primeiro a ser retirado da estrutura.

Podemos fazer uma comparação com uma pilha de pratos, em que, se quisermos adicionar um prato na pilha, devemos colocá-lo topo, e, para pegar um prato da pilha, retiramos o do topo. Dessa forma, temos que retirar o prato do topo para ter acesso ao próximo prato. Portanto, essa manipulação é feita apenas por uma das extremidades da lista, pelo topo.

O objetivo deste trabalho é corrigir e rodar o código a partir do exemplo definido no conteúdo deste material.

2 CÓDIGO PROPOSTO

As imagens a seguir descrevem a proposta de um código para ser corrigido, compilado e na sequência demonstrar na prática a teoria sobre a Pilhas.

Trabalho

Continuação do códigos dos exemplos de pilha

```
while( 1 ){ /* loop infinito */
    printf("n1- empilhar (push)\n");
    printf("2- desempilhar (POP)\n");
    printf("3- Mostrar o topo\n");
    printf("4- sair\n");

    printf("nopcao? ");
    scanf("%d", &op);

    switch (op){
        case 1: //push
            if( estacheia( &minhapilha ) == 1 )
                printf("nPILHA CHEIA!\n");
            else {
                printf("nVALOR? ");
                scanf("%f", &valor);
                empilhar (&minhapilha, valor);
            }
            break;

        case 2: //pop
            if ( !estavazia(&minhapilha) == 1 )
                printf( "nPILHA VAZIA!\n" );
            else{
                valor = desempilhar (&minhapilha);
                printf ( "n%1f DESEMPILHADO\n", valor );
            }
            break;

        case 3: // mostrar o topo
            if ( !estavazia (&minhapilha) == 1 )
                printf( "nPILHA VAZIA!\n" );
            else {
                valor = retomatopo (&minhapilha);
                printf ( "nTOPO: %1f\n", valor );
            }
            break;

        case 4:
            exit(0);
        default: printf( "nOPCAO INVALIDA!\n" );
    }
}
```



Trabalho

Criar e fazer rodar os códigos dos exemplos de pilha

```
#include
struct Pilha {
    int topo; /* posição elemento topo */
    int capa;
    float *pElem;
};

void criarpilha( struct Pilha *p, int c ){
    p->topo = -1;
    p->capa = c;
    p->pElem = (float*) malloc ( c * sizeof(float));
}

int estavazia ( struct Pilha *p ){
    if( p-> topo == -1 )
        return 1; // true
    else
        return 0; // false
}

int estacheia ( struct Pilha *p ){
    if ( p->topo == p->capa - 1 )
        return 1;
    else
        return 0;
}

void empilhar ( struct Pilha *p, float v ){
    p->topo++;
    p->pElem [p->topo] = v;
}

float desempilhar ( struct Pilha *p ){
    float aux = p->pElem [p->topo];
    p->topo--;
    return aux;
}

float retomatopo ( struct Pilha *p ){
    return p->pElem [p->topo];
}

int main(){
    struct Pilha minhapilha;
    int capacidade, op;
    float valor;

    printf( "nCapacidade da pilha? " );
    scanf( "%d", &capacidade );

    criarpilha (&minhapilha, capacidade);
}
```



3 CORREÇÃO DO CÓDIGO

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

struct Pilha{

    int topo;/* posição elemento topo */
    int capa;
    float *pElem;

};

void criarpilha( struct Pilha *p, int c){

    p->topo = -1;
    p->capa = c;
    p->pElem = (float*) malloc (c*sizeof(float));

}

int estavazia ( struct Pilha *p){

    if(p-> topo ==-1)

        return 1; //true

    else

        return 0; //false

}

int estacheia (struct Pilha *p){

    if (p->topo == p->capa -1)
```

```
        return 1;

    else

        return 0;

}

void empilhar ( struct Pilha *p, float v){

    p->topo ++;
    p->pElem [p->topo] = v;

}

float desempilhar (struct Pilha *p){

    float aux = p->pElem [p->topo];
    p->topo--;
    return aux;

}

float retornatopo ( struct Pilha *p){

    return p->pElem [p->topo];

}

int main(){

    struct Pilha minhapilha;
    int capacidade, op;
    float valor;

    printf( "\nCapacidade da pilha?" );
    scanf( "%d", &capacidade );
```

```
criarpilha (&minhapilha, capacidade);
```

```
while(1){/* loop infinito */
```

```
    printf("\n1- empilhar (push)\n");  
    printf("2- desempilhar (POP)\n");  
    printf("3- Mostrar o topo \n");  
    printf("4- sair\n");
```

```
printf("\nopcao?");  
scanf("%d", &op);
```

```
switch (op){
```

```
    case 1: //push
```

```
        if( estacheia( &minhapilha ) == 1)
```

```
            printf("\nPILHA CHEIA!\n");
```

```
        else{
```

```
            printf("\nVALOR?");  
            scanf("%f", &valor);  
            empilhar (&minhapilha, valor);
```

```
        }
```

```
        break;
```

```
    case 2: //pop
```

```
        if ( estavazia(&minhapilha) == 1 )
```

```
            printf( "\nPILHA VAZIA! \n");
```



```
else{

    valor = desempilhar (&minhapilha);
    printf ( "\n%.1f DESEMPILHADO!\n", valor );

}
break;

case 3: //mostrar o topo
    if ( estavazia (&minhapilha) == 1)

        printf( "\nPILHA VAZIA!\n" );

    else{

        valor = retornatopo (&minhapilha);
        printf ( "\nTOPO: %.1f\n", valor );

    }
    break;

case 4:

    exit(0);

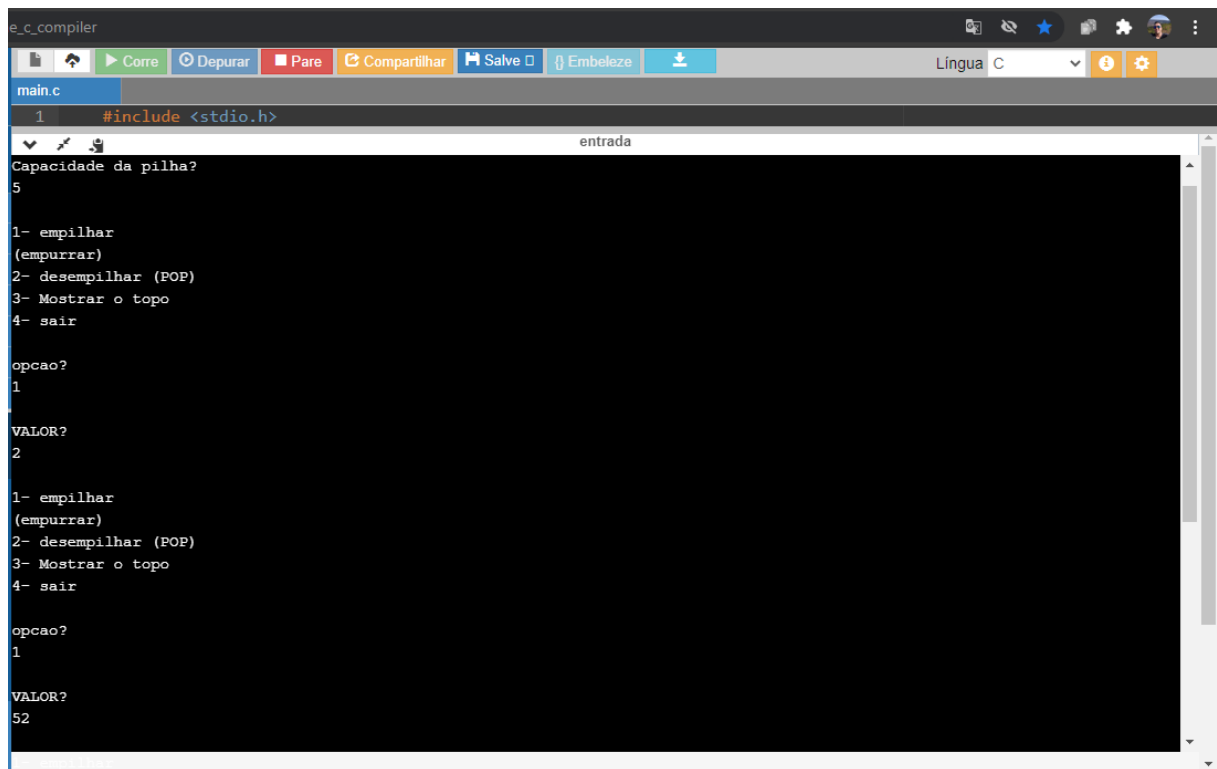
default: printf("\nOPCAO INVALIDA! \n");

}

}

}
```

4 COMPILAÇÃO OBTIDA



```
e_c_compiler
main.c
1 #include <stdio.h>

Capacidade da pilha?
5

1- empilhar
(empurrar)
2- desempilhar (POP)
3- Mostrar o topo
4- sair

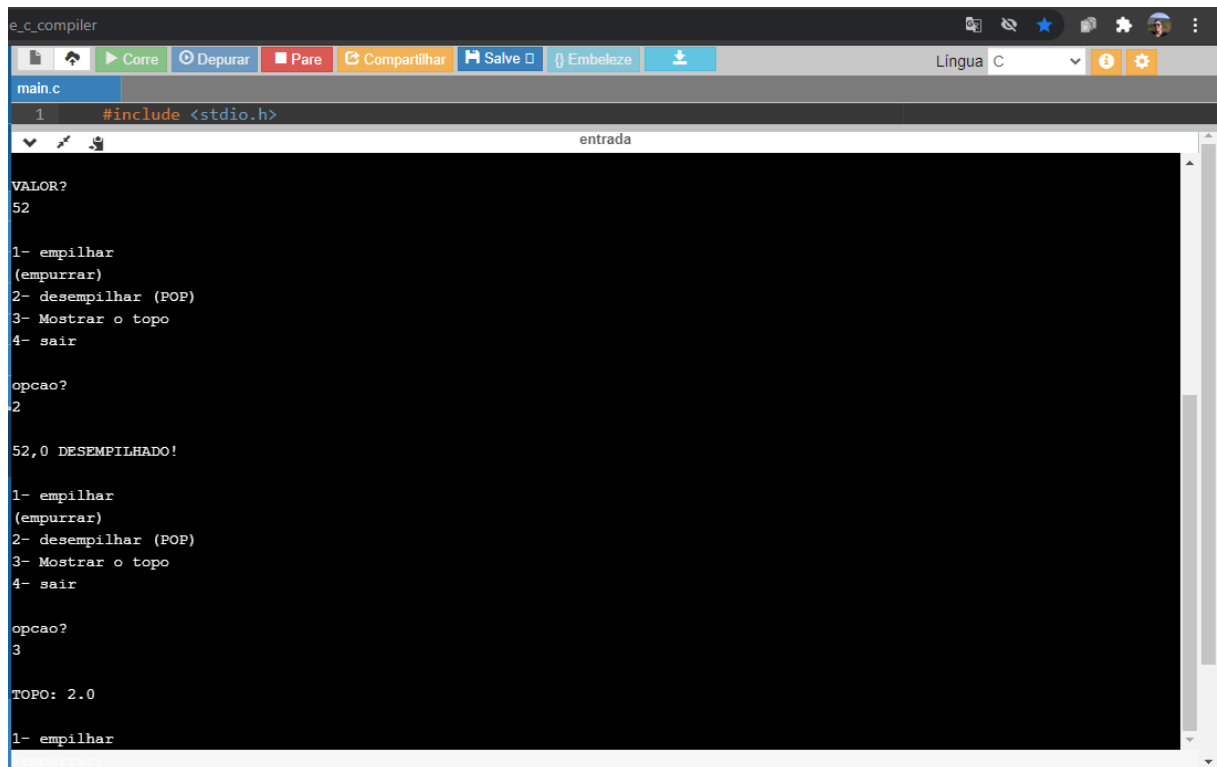
opcao?
1

VALOR?
2

1- empilhar
(empurrar)
2- desempilhar (POP)
3- Mostrar o topo
4- sair

opcao?
1

VALOR?
52
```



```
e_c_compiler
main.c
1 #include <stdio.h>

VALOR?
52

1- empilhar
(empurrar)
2- desempilhar (POP)
3- Mostrar o topo
4- sair

opcao?
2

52,0 DESEMPILHADO!

1- empilhar
(empurrar)
2- desempilhar (POP)
3- Mostrar o topo
4- sair

opcao?
3

TOPO: 2.0

1- empilhar
```

Capacidade da pilha? 5

- 1- empilhar (empurrar)
- 2- desempilhar (POP)
- 3- Mostrar o topo
- 4- sair

opcao? 1

VALOR? 2

- 1- empilhar (empurrar)
- 2- desempilhar (POP)
- 3- Mostrar o topo
- 4- sair

opcao? 1

VALOR? 52

- 1- empilhar (empurrar)
- 2- desempilhar (POP)
- 3- Mostrar o topo
- 4- sair

opcao? 2

52,0 DESEMPILHADO!

- 1- empilhar (empurrar)
- 2- desempilhar (POP)
- 3- Mostrar o topo
- 4- sair

opcao? 3

TOPO: 2.0

5 CONCLUSÃO

A trabalho evidenciou a falta da Declaração da Interfaces (ou Assinaturas), após a declaração das funções e a indentação do código, o mesmo foi compilado no site: https://www.onlinegdb.com/online_c_compiler, onde foi percebido através do programa principal e manipulação dos dados na pilha a execução das operações de: criação (pull), inserção (push), remoção (pop) e acesso ao elemento (top).

REFERÊNCIAS

CCM. Como fazer pilhas na linguagem C. <https://br.ccm.net/faq/10218-como-fazer-pilhas-na-linguagem-c>. Acesso em: 08.04.2021

Estrutura de dados [recurso eletrônico] / Adriana de Souza Vetorazzo...[et al]; [revisão técnica: Jeferson Faleiro Leon de Souza Machado]. – Porto Alegre: SAGAH, 2018.