Bacharelado sistemas de Informação - 2 semestre

Disciplina: Estrutura de Dados

Aluno: Vinícius da Silva de Souza

Lista de Exercícios

## **Lista de exercícios de programação em linguagem C – Pilhas**

**Exercício 1 – Conceito de Pilha**

Explique o que é uma pilha em Estruturas de Dados. Dê dois exemplos do cotidiano em que o comportamento de uma pilha pode ser observado.

### Uma pilha é uma estrutura de dados onde o último elemento inserido é o primeiro a ser removido. As operações principais são inserir (push) e remover (pop) sempre no topo da pilha.

### Dois exemplos do cotidiano que seguem esse princípio são:

### Livros empilhados: para pegar o primeiro livro que colocou, você precisa remover os de cima primeiro; A pilha de abas no navegador: A última aba aberta é a primeira a ser fechada.

**Exercício 2 – Estruturas Estática x Dinâmica**

a) Diferencie pilha estática e pilha dinâmica.

Uma pilha pode ser implementada de forma estática ou dinâmica.

Já a pilha dinâmica é implementada por meio de alocação encadeada, geralmente com ponteiros, permitindo que o tamanho cresça ou diminua conforme os elementos são inseridos ou removidos.

b) Cite vantagens e desvantagens de cada abordagem.

A pilha estática utiliza um vetor com tamanho fixo já definido , tendo como vantagem maior simplicidade na implementação e acesso direto aos elementos, mas como desvantagem o fato de estar limitada a um tamanho máximo, podendo faltar espaço mesmo que a memória total do sistema ainda comporte mais dados.

A pilha dinâmica tem sua principal vantagem a flexibilidade de tamanho, porém ela apresenta a desvantagem de ser mais complexa de implementar e possuir um custo adicional de gerenciamento de memória.

**Exercício 3 –** Considere uma estrutura de dados em uma tabela denominada TAB.  
 A posição 1 armazena a próxima posição livre da tabela. Em dado momento, a tabela apresenta os dados preenchidos, conforme consta abaixo:

Tabela TAB

7 25 12 21 18 23

1 2 3 4 5 6 7 8 9 10

A inserção do número 15 na tabela resultou no exposto abaixo:

Tabela TAB

8 15 25 12 21 18 23

1 2 3 4 5 6 7 8 9 10

Ao serem retirados dois elementos, a tabela apresenta a seguinte configuração:

Tabela TAB

6 12 21 18 23

1 2 3 4 5 6 7 8 9 10

Tal estrutura é uma?

a) fila com início na posição 2.  
 b) fila com início na posição TAB(1).  
 c) fila com fim na posição 2.  
 d) pilha com topo na posição TAB(1).  
 e) pilha com topo na posição 2.

**Exercício 4 – Operações Fundamentais**

Liste e explique as operações básicas de uma pilha:

· Push - Insere (empilha) um novo elemento no topo da pilha. Se for pilha estática, verifica antes se não está cheia.

· Pop - Remove (desempilha) o elemento do topo da pilha. Se estiver vazia, a operação não é permitida.

· Topo (ou peek) - Retorna o valor do elemento do topo, sem removê-lo da pilha.

· isEmpty - Verifica se a pilha está vazia (sem elementos). Retorna verdadeiro ou falso.

· isFull - Verifica se a pilha está cheia (no caso da pilha estática com tamanho fixo).

**Exercicio 5 – Pilha Estática**

Implemente uma pilha estática utilizando vetor em C. O programa deve com ter as funções básicas (push, pop, topo, isEmpty, isFull) e um menu para o usuário interagir.

#include <stdio.h>

#include <stdlib.h>

#define TAM 5

struct Pilha {

int dados[TAM];

int topo;

};

void inicializar(struct Pilha \*p) {

p->topo = -1;

}

int isEmpty(struct Pilha \*p) {

return (p->topo == -1);

}

int isFull(struct Pilha \*p) {

return (p->topo == TAM - 1);

}

void push(struct Pilha \*p, int valor) {

if (isFull(p)) {

printf("A pilha está cheia! Não é possível empilhar.\n");

} else {

p->topo++;

p->dados[p->topo] = valor;

printf("Valor %d empilhado com sucesso!\n", valor);

}

}

void pop(struct Pilha \*p) {

if (isEmpty(p)) {

printf("A pilha está vazia! Nada para desempilhar.\n");

} else {

int removido = p->dados[p->topo];

p->topo--;

printf("Valor %d removido da pilha.\n", removido);

}

}

void topo(struct Pilha \*p) {

if (isEmpty(p)) {

printf("A pilha está vazia!\n");

} else {

printf("Topo da pilha: %d\n", p->dados[p->topo]);

}

}

void mostrar(struct Pilha \*p) {

if (isEmpty(p)) {

printf("A pilha está vazia!\n");

} else {

printf("Elementos da pilha:\n");

for (int i = p->topo; i >= 0; i--) {

printf("| %d |\n", p->dados[i]);

}

}

}

int main() {

struct Pilha p;

inicializar(&p);

int opcao, valor;

do {

printf("\n=== MENU PILHA ===\n");

printf("1 - Empilhar (push)\n");

printf("2 - Desempilhar (pop)\n");

printf("3 - Mostrar topo\n");

printf("4 - Mostrar pilha\n");

printf("5 - Sair\n");

printf("Escolha uma opcao: ");

scanf("%d", &opcao);

switch (opcao) {

case 1:

printf("Digite um valor: ");

scanf("%d", &valor);

push(&p, valor);

break;

case 2:

pop(&p);

break;

case 3:

topo(&p);

break;

case 4:

mostrar(&p);

break;

case 5:

printf("Saindo do programa...\n");

break;

default:

printf("Opcao invalida!\n");

}

} while (opcao != 5);

return 0;

}