UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS CRATEÚS
SISTEMAS DE INFORMAÇÃO
DOCENTE: RAFAEL MARTINS BARROS
DISCIPLINA: ESTRUTURA DE DADOS

DISCENTES: ANA JAZA TAVARES SOARES - 558540, ANA LETICIA DE OLIVEIRA SOARES - 555346.

RELATÓRIO: PILHAS E FILAS

CRATEÚS
2024

SUMÁRIO

CRATEÚS
2024

1. Objetivo

O objetivo deste projeto é implementar duas estruturas de dados abstratas (TADs) fundamentais, a Pilha de Números Inteiros e a Fila de Pilhas, utilizando a linguagem C, aplicando os conceitos de estrutura de dados. O projeto foi desenvolvido em dupla por Jaza e Ana Letícia, utilizando uma abordagem incremental, com versionamento por meio de commits no GitHub.

A Pilha de Números Inteiros deverá seguir o princípio LIFO (Last In, First Out), enquanto a Fila de Pilhas deve combinar conceitos de fila (FIFO - First In, First Out) e pilha, onde cada elemento da fila será uma pilha de números inteiros. Além disso, o projeto visa desenvolver uma interface de usuário baseada em texto que permita a interação com as estruturas de dados, permitindo ao usuário realizar as operações de inserção, remoção, e consulta de elementos nas pilhas e na fila de pilhas. Este projeto busca consolidar conceitos de manipulação de ponteiros e gerenciamento de memória, enquanto garante o tratamento adequado de erros para operações em estruturas vazias, promovendo um entendimento mais profundo das abstrações de dados e suas aplicações práticas.

2. Principais funções implementadas e Big-O

2.1.Pilha:

2.1.1 Criar Pilha

Inicializa uma nova pilha vazia. O(1)

2.1.2. Empilhar:

Insere um elemento no topo da pilha. O(1)

2.1.3. Desempilhar:

Remove e retorna o elemento no topo da pilha. O(1)

2.1.4. Verificar Topo:

Retorna o elemento do topo sem removê-lo. O(1)

2.1.5. Verificar se a Pilha está Vazia:

Retorna verdadeiro se a pilha estiver vazia. O(1)

2.1.6 Tamanho da Pilha:

Retorna o número de elementos na pilha. O(1)

2.2. Fila:

2.2.1. Criar Fila:

Inicializa uma nova fila vazia. O(1)

2.2.2.Enfileirar:

Insere um elemento no final da fila. O(1)

2.2.3 Desenfileirar:

Remove e retorna o elemento no início da fila. O(1)

2.2.4. Verificar Primeiro Elemento:

Retorna o primeiro elemento da fila sem removê-lo. O(1)

2.2.5. Verificar se a Fila está Vazia:

Retorna verdadeiro se a fila estiver vazia. O(1)

2.2.6. Tamanho da Fila:

Retorna o número de elementos na fila. O(1)

Todas as operações são O(1) porque a estrutura olha para um elemento fixo, o último elemento no caso da pilha ou primeiro elemento no caso da fila, ou seja, o índice é bem definido. Além disso, a verificação de tamanho é O(1) porque depende de uma variável já calculada dentro da struct.

3. Como o trabalho foi organizado

Para a realização deste trabalho, formamos um grupo na plataforma de mensagens WhatsApp focado em discutir e buscar soluções para cada tarefa solicitada. Com o objetivo de manter uma boa organização, divisão de responsabilidades e divisão das implementações. Jaza ficou responsável pelas quatro primeiras funções, enquanto Ana Letícia implementou as demais funções, além da parte de criação do menu de interação com o usuário.

4. Principais dificuldades

As principais dificuldades encontradas ao implementar este código foram compreender como as diferentes partes do código interagiam entre si e gerenciar

corretamente os ponteiros, especialmente no que diz respeito à manipulação de pilhas dentro dos nós da fila. Tal estrutura adicionou um nível extra de complexidade ao projeto. Além disso, as funções de desempilhamento e remoção de elementos também foram bastante desafiadoras. Contudo, esses obstáculos foram superados por meio de estudos aprofundados nos slides, pesquisas adicionais e com o apoio de colegas externos à turma.

5. Conclusão

A implementação das estruturas de dados de pilha e fila neste projeto nos permitiu consolidar os conceitos fundamentais de manipulação de ponteiros e abstração de tipos de dados em C. Apesar das dificuldades iniciais, como o correto gerenciamento de ponteiros, especialmente em uma estrutura de filas contendo pilhas, conseguimos superar esses desafios por meio de discussões em grupo, pesquisas adicionais e estudo aprofundado. O aprendizado adquirido ao enfrentar e solucionar os problemas, sobretudo nas operações de remoção e desempilhamento, foi extremamente valioso e contribuiu para o fortalecimento dos conceitos de estrutura de dados. Ao final, todas as operações apresentaram complexidade O(1), mostrando que o código foi otimizado e eficiente.