Trabalho Prático 1 – Estrutura de Dados

Chrystian Paulo Ferreira de Melo – 2020089879

1. Introdução

O trabalho surgiu da necessidade de solucionar um problema na segurança durante a comunicação de computadores. Para solucionar o problema foi utilizado o conceito de fila, uma estrutura de dados que possibilita tratar as informações da melhor forma possível.

O desenvolvimento foi feito na linguagem C++ pela facilidade da criação de códigos mais limpos, modularizados e organizados. Além disso, o ambiente utilizado foi Visual Studio 2019 no sistema Windows 10.

2. Instruções de compilação e execução

Para compilação do programa, o usuário deve abrir o terminal na pasta tp e digitar make.

Em seguida, será possível executar entrando na pasta bin, clicando no arquivo executável nomeado “run.out”. Após isso digite o caminho do arquivo de entrada no terminal aberto pelo executável.

3. Implementação

Para resolver o problema proposto, foram criadas duas classes: Node (Nodo da fila) e Queue(Fila de nodos).

Na classe Node temos as seguintes funções:

- Node(): Constrói a classe, com dados padrões ou dados especificados na chamada.
- getContent(): Retorna o conteúdo do nodo atual.
- getNext(): Retorna o próximo nodo.
- setNext(): Determina qual o próximo nodo na fila.
- edit(): Edita o conteúdo do nodo atual.
- hasNext(): Verifica se existe um nodo após.
- connections(): Verifica quantas nodos tem após.
- isEmpty(): Verifica se o nodo tem algum dado ou está vazio.

Na classe Queue temos as seguintes funções:

- Queue(): Constrói a classe, com dados padrões ou dados especificados na chamada.
- getNode(): Retorna o primeiro nodo da fila.
- push(): Adiciona um novo nodo na fila, respeitando a política FIFO.
- pop(): Remove o primeiro nodo na fila.

- setOnTop(): Coloca o nodo especificado na primeira posição da fila.
- print(): Imprime na tela todos os elementos da fila, respeitando a ordem que os nodos foram adicionados.

4. Análise de complexidade

Na classe Node:

- Node(): O(1)
- getContent(): O(1)
- getNext(): O(1)
- setNext(): O(1)
- edit(): O(1)
- hasNext(): O(1)
- connections(): O(n)
- isEmpty(): O(1)

Na classe Queue:

- Queue(): O(1)
- getNode(): O(1)
- push(): O(n)
- pop(): O(1)
- setOnTop(): O(n)
- print(): O(n)

A complexidade de espaço do programa é constante.

5. Conclusão

O resultado foi um sistema que respeita as regras especificadas e cumpre o objetivo proposto. Não houve desenvolvimento de lógicas que fogem do que foi proposto inicialmente, como tratamento de erros de entrada.

6. Referências

1. Material de aula (pdf e videos)
2. stackoverflow.com
3. www.cplusplus.com
4. www.tutorialspoint.com
5. en.cppreference.com
6. www.geeksforgeeks.org
7. en.wikipedia.org
8. www.learncpp.com