Universidade Federal do Piaui – UFPI
Campus Senador Helvídio Nunes de Barros –CSHNB
Curso de Sistemas de Informação /Bloco: IV
Disciplina: Estruturas de Dados II
Professora: Juliana Oliveira de Carvalho
Acadêmicos: Vinicius da Silva Nunes

RELATÓRIO DE TRABALHO DE ESTRUTURA DE DADOS 2

Vinicius da Silva Nunes

01 maio 2025

Resumo do Projeto:

Este projeto tem o objetivo de discorrer sobre a resolução dos problemas envolvendo a Estrutura de Dados da **Árvore Binária de Busca** e **Árvore Binária de Busca AVL**. As resoluções foram implementadas usando a linguagem de programação C, a linguagem C é uma linguagem considerada nível intermediário, pois combina aspectos das linguagem de máquina. Estrutura de Dados é o modo de armazenar,acessar e organizar dados em um sistemas de maneira eficaz e otimizada, esse tipo de conceito é essencial para os profissionais da área da computação que estão em constante contato com diferentes formas de Estrutura de Dados. Os principais tipos de Estrutura de Dados na computação são, **arrays**, **grafos**, **listas simples** e **listas encadeadas**, **filas**, **pilhas** e **árvores**. A seguir serão abordados assuntos que envolvem as Estruturas de Dados da **Árvore Binária de Busca** e **Árvore Binária de Busca AVL** como a solução de um exercício proposto em sala de aula.

Introdução:

Como citado anteriormente as estruturas de dados tem um papel fundamental na área da computação principalmente as que envolvem os assuntos como árvores, a existência de várias formas de implementação desse tipo de estrutura em computação torna possível não só resolver problemas complexos, mas também realizar comparações de tempo e execução de programas que implementam esse tipo de estrutura. Com isso este relatório tem a finalidade de descrever sobre as implementações de das estrutura de dados árvores e seus tempos de medições de tempo de inserção em buscar e apresentar resultados que mostram a diferença de ambas as implementações.

Seções Específicas:

Sobre o problema

O problema apresentado possui os seguintes requisitos: Faça um programa em C .Artista. E para cada artista deve ser ter o nome do artista, tipo (cantor(a), Dupla, Banda, Grupo, ...), o estilo musical, o número de álbuns, e os Álbuns (endereço da árvore binária para Álbuns). Cada álbum deve ter, o título, o ano de lançamento, a quantidade de músicas e as Músicas (endereço da árvore binária para Músicas). Para cada música deve se ter o título, e a quantidade de minutos. Quando o usuário executar o'programa o mesmo deve permitir:

i)Cadastrar Artista: cadastrar os dados dos artistas organizados em uma árvore binária pelo nome do artista a qualquer momento, o usuário pode cadastrar um artista a qualquer momento, não permita cadastro repetido.

ii)Cadastrar Álbuns: cadastrar os dados de Álbuns de um determinado artista organizados em uma árvore binária pelo título do álbum a qualquer momento, lembre-se um álbum só pode ser cadastrado para um artista já castrado e o álbum não pode se repetir para um mesmo artista

iii)Cadastrar Músicas: cadastrar as músicas de um álbum de um artista em uma árvore binária organizada pelo título, lembre-se uma música só pode ser cadastrada para um álbum que já existe e a música não pode se repetir para um mesmo álbum.

iv)Mostrar todos os artistas cadastrados.v)Mostrar todos os artistas cadastrados de um determinado tipo.vi)Mostrar todos os artistas cadastrados de um determinado estilo musical.vii)Mostrar todos os artistas cadastrados de um determinado estilo musical e tipo de artista. viii)Mostrar todos os álbuns de um determinado artista. ix)Mostrar todos os álbuns de um determinado ano de um artista.x) Mostrar todas as músicas de um determinado álbum de um determinado artista.xi)Mostrar todos os álbuns de um determinado ano de todos os artistas cadastrados xii)Mostrar os dados de uma determinada Música (usuário entrar com o título da música): nome artista,título do álbum, ano de lançamento. xiii)Criar uma árvore binária de playlists, na qual cada playlist contém um nome (organização da árvore), e uma árvore binária de músicas. Não permita playlist com títulos iguais. As músicas das playlists devem estar cadastradas na árvore de músicas de um álbum de um artista. A árvore de música da playlist deve conter: nome do artista, título do álbum e o título da Música.xiv)Mostrar todos os dados de uma determinada playlist.xv)Permita remover uma música de uma determinada playlist. xvi)Permita remover uma playlist, lembre-se de remover a árvore binária da playlist também .xvii)Permite remover uma música de um determinado álbum de um determinado artista. Lembre-se só poderá ser removida se não fizer parte de nenhuma playlist.

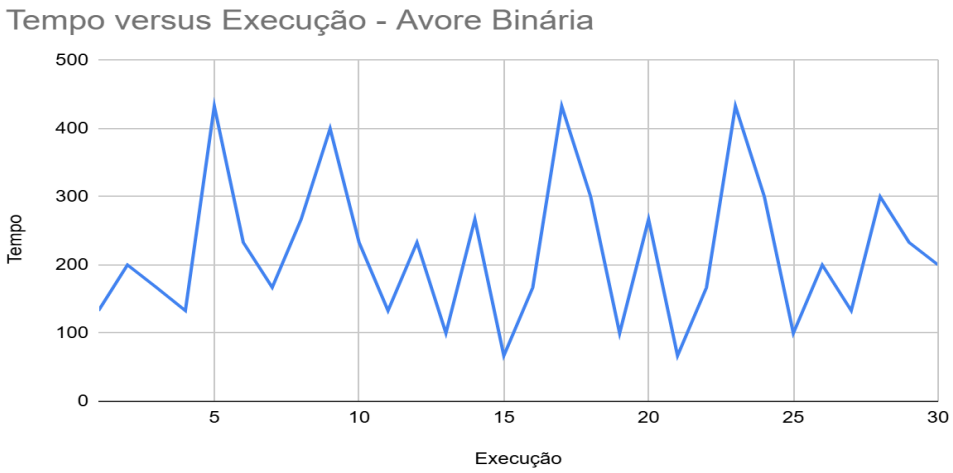
Sobre a solução

Para a resolução do problema foi necessário criar TADs(Tipos Abstratos de Dados) para armazenar os dados solicitados na questão como,,Árvore de Artista, Árvore de Álbum, Música e Playlist e a criação de menu de opções para o usuário.Para implementação da **árvore binária AVL** que segue os mesmos parâmetros de criação, inserção, remoção e busca foi necessário a criação de algoritmos adicionais como os **algoritmos de balanceamento**, **rotações para direita e esquerda** (essas podendo ser duplas dependendo do momento da inserção e remoção na árvore).

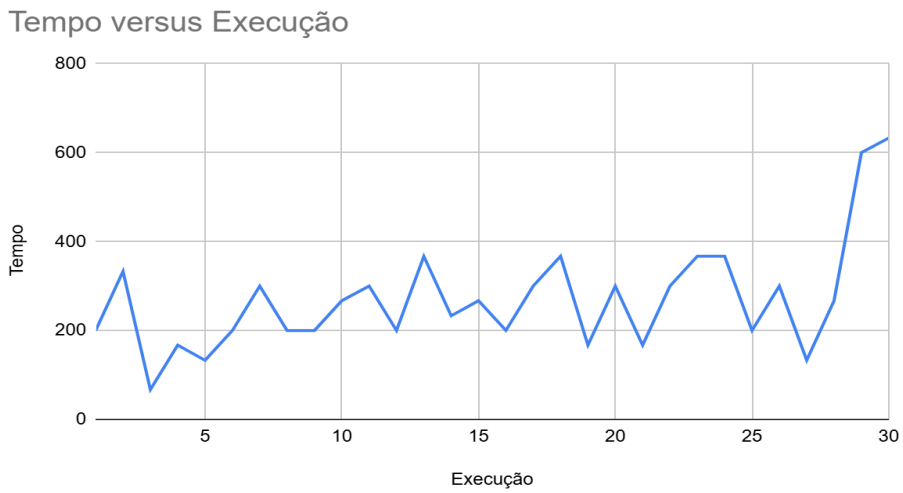
Essas funções adicionais foram necessárias pois ao contrário da árvore binária simples, a árvore binária AVL é uma árvore que tem que estar perfeitamente balanceada, esse balanceamento é realizado sempre em fases de inserção e remoção de dados na árvore utilizando o **fator de balanceamento** dos nós à esquerda e à direita das suas subárvores. O fator de balanceamento é obtido da diferença entre a altura nó da subárvore à esquerda e a altura da subárvore à direita.

Resultados da Execução do Programa:

Com a resolução do programa concluída, foi possível fazer experimentos como solicitado na questão 2 do exercício, exceto a busca solicitada na letra b utilizando o item 1 da questão 1. Os experimentos foram realizados em um notebook, com as seguintes configurações: processador Core i3, 20 GB de memória RAM, SSD 240 GB, Sistema operacional Windows 11. Os experimentos de operação de inserção na de cada item na árvore de curso para obter um dado mais preciso e possibilitar uma comparação mais técnica tanto na árvore binária de busca quanto na árvore binária avl. Os dados obtidos nos testes são apresentados nos gráficos abaixo.



Foram realizadas 10 execuções em cada teste de inserção com 3000 elementos para inserir na árvore de curso. O tempo foi obtido em milissegundos, como mostra os gráficos acima. Os dados foram extraídos dos testes obtidos na árvore binária de busca. Em seguida realizou-se testes de inserção na árvore AVL, que é uma variação da árvore binária de busca, com funções adicionais de balanceamento e rotações para manter a árvore balanceada. O dados dos testes podem ser visto nas tabela abaixo:



Conclusão:

Como já citado anteriormente, o experimento teve como objetivo realizar a comparação de média de tempo na inserção e busca de dados nas duas árvores(binária e Avl), no entanto é válido ressaltar que por motivos de falhas na implementação da função de busca, não foi possível realizar o teste de tempo nessas operações.

Porém levando em consideração as médias de tempo nas tabelas 1 e 2 apresentadas na parte Resultados e execuções do programa o tempo de inserção na árvore binária relativamente mais rápido com o tempo variando enquanto na Avl o bem mais demorado que na versão com a binária. Isso se dá pelo fato de que na árvore o balanceamento é realizado no momento da inserção, o que pode ser mais demorado, já que o algoritmo tem que analisar toda a subárvore e realizar o balanceamento quando necessário.

As lições aprendidas durante a resolução do trabalho, foram aprimorar o conhecimento nas estruturas de árvores, implementação recursiva, endereçamento de ponteiros e também a realizar experimentos por meios de algoritmos automatizados que realizavam os testes apresentados neste relatório.