Trabalho Prático – Árvores e Aplicações em Estruturas de Dados

1. Objetivo

Desenvolver um **sistema prático para manipulação, armazenamento e busca de dados** utilizando **estruturas de árvores**, aplicando conceitos de **eficiência, balanceamento e complexidade de algoritmos**.

O aluno deverá **implementar e comparar** diferentes tipos de **árvores de busca** — **Binária, AVL e Rubro-Negra** — analisando seu comportamento nas operações de **inserção, remoção e busca**, com **geração automática de dados** para os testes.

Tempo estimado: **20 horas de desenvolvimento.**

2. Formação de Grupos e Apresentação

O trabalho deverá ser desenvolvido **em grupos de até 4 integrantes**.
Cada grupo será responsável por toda a implementação, documentação e demonstração prática do sistema.

Ao término do prazo de desenvolvimento, será realizada uma **apresentação obrigatória**, com duração de **10 a 15 minutos**, na qual o grupo deverá:

- Explicar brevemente a **teoria das árvores** implementadas (binária, AVL e rubro-negra);
- Demonstrar **o funcionamento prático do sistema** (inserção, busca, remoção e balanceamento);
- Exibir os **resultados comparativos de desempenho** entre as estruturas;
- Responder a eventuais questionamentos sobre a implementação e os conceitos teóricos.

A apresentação poderá ser feita com o código em execução, slides ou ambos, desde que o **sistema em funcionamento** seja mostrado.

3. Cenário / Tema do Projeto

O trabalho propõe o desenvolvimento de um sistema capaz de **armazenar e gerenciar elementos de forma estruturada em árvores**, com o objetivo de **comparar o desempenho** entre as abordagens de árvores tradicionais e balanceadas.

Cada elemento armazenado conterá, no mínimo:

- Um **identificador numérico único (ID)**;
- Um **nome ou valor genérico (string)**;
- (Opcional) um campo adicional, como categoria ou peso, para enriquecer os testes.

Os dados deverão ser **gerados automaticamente** pelo próprio programa, com a criação de listas de elementos **aleatórios** de tamanhos variados (por exemplo, 100, 1.000 e 10.000 registros).

4. Requisitos Técnicos do Projeto

4.1 Estruturas de Árvores (Obrigatórias)

O grupo deverá **implementar e testar** os seguintes tipos de árvores:

1. **Árvore Binária de Busca (BST)**
    - Operações de inserção, busca e remoção;
    - Impressões in-order, pre-order e post-order.
2. **Árvore AVL (Auto-balanceada)**
    - Implementação de rotações simples e duplas (direita e esquerda);
    - Cálculo e ajuste do fator de balanceamento após cada operação.
3. **Árvore Rubro-Negra (Red-Black Tree)**
    - Implementação das propriedades de coloração e regras de balanceamento;
    - Correta aplicação de rotações e trocas de cor.

4.2 Operações Mínimas Requeridas

Cada estrutura deverá permitir:

- Inserir um novo elemento (com ID gerado automaticamente);
- Remover um elemento específico;
- Buscar um elemento pelo ID;
- Exibir a árvore ordenada (in-order traversal);
- Exibir métricas:
    - Altura da árvore;
    - Número de nós;
    - Quantidade de rotações realizadas (AVL e Rubro-Negra).

4.3 Comparação de Desempenho

O sistema deve executar **testes automáticos** comparando o desempenho entre as três estruturas.

Para isso, deverá:

1. **Gerar automaticamente N elementos** (N = 100, 1.000 e 10.000, por exemplo);
2. Inserir todos os elementos em cada tipo de árvore;
3. Executar operações de busca e remoção;
4. Medir:

- Tempo total de execução (em milissegundos);
- Altura final da árvore;
- Número médio de comparações por operação.

Os resultados devem ser exibidos em formato de **tabela ou gráfico comparativo**.

5. Relatório Técnico (Obrigatório)

O relatório deve conter:

- **Introdução teórica** sobre árvores binárias, AVL e rubro-negras;
- **Descrição das implementações** e principais funções;
- **Apresentação dos resultados experimentais** (tabelas ou gráficos);
- **Análise de complexidade** de cada tipo de árvore (melhor, médio e pior caso);
- **Conclusão comparativa**: qual estrutura apresentou melhor desempenho e em quais situações;
- Prints do código e da execução do programa.

Formato: **PDF (mínimo 3 e máximo 6 páginas).**

6. Linguagens e Ferramentas Permitidas

O trabalho poderá ser desenvolvido em uma das seguintes linguagens:

- **Python** (recomendado, utilizando classes e estruturas recursivas)
- **Java** (com orientação a objetos e uso de genéricos)
- **C / C++** (com ponteiros e structs)
- **C#**

Não é permitido o uso de bibliotecas prontas de árvores.
Todo o código deve ser implementado manualmente pelo grupo.

7. Entregáveis

Cada grupo deverá entregar um pacote contendo:

- Código-fonte completo e comentado (`.zip`);
- Relatório técnico em PDF;
- Prints de execução e/ou vídeo curto demonstrativo;
- (Opcional) Arquivo `.csv` com resultados dos testes comparativos.

8. Dicas Práticas

- Utilize **funções recursivas** para percorrer as árvores.

- Gere dados automaticamente com funções de randomização (`random`, `rand`, etc.).
- Registre o **tempo de execução** e o **número de comparações** em cada operação.
- Comece pela árvore binária de busca e evolua gradualmente para AVL e Rubro-Negra.
- Teste casos de inserção ordenada (pior caso) e aleatória (caso médio).
- Documente cada função e mantenha o código modularizado.