**UNIVERSIDADE FEDERAL DE RORAIMA**

**CENTRO DE CIÊNCIAS E TECNOLOGIA**

**DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO**

**CURSO DE CIÊNCIA DA COMPUTAÇÃO**

**ESTRUTURA DE DADOS II**

**DOSCENTE: JEAN BERTRAND PAIXÃO DA SILVA**

**DISCENTE: LUCAS GABRIEL ROCHA CONSTANCIO**

**TRABALHO PRÁTICO II – Árvore binaria de busca e arvore AVL**

**Relatório – Estrutura de Dados II**

Tema: Implementação de Árvores Binárias de Busca (ABB) e Árvores AVL

**Objetivo**

Implementar uma **Árvore Binária de Busca (ABB)** e uma **Árvore AVL**, comparando seus comportamentos em termos de inserção, percursos e busca, verificando também o desempenho de cada uma.

**Descrição das Estruturas**

**Árvore Binária de Busca (ABB)**

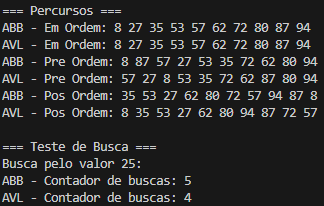
* Estrutura de árvore onde, para cada nó:
  + Os valores menores ficam à esquerda;
  + Os valores maiores ficam à direita.
* Não garante balanceamento automático.
* Busca tem complexidade:
  + Melhor caso: O(log n);
  + Pior caso (desbalanceada): O(n).

**Árvore AVL**

* É uma árvore binária de busca com balanceamento automático.
* Após cada inserção ou remoção, a árvore verifica o **fator de balanceamento** e realiza rotações se necessário.
* Mantém a altura da árvore aproximadamente logarítmica.
* Busca tem complexidade garantida de O(log n).

**Dados Utilizados e Percusos Realizados**

Os valores são gerados de forma aleatória utilizando o srand para gerar 10 valores de 0 a 99:



**Observação:**

* No percurso **em ordem**, ambas as árvores mostram os dados ordenados crescentemente.
* Os percursos **pré-ordem** e **pós-ordem** apresentam diferenças claras na disposição dos nós, principalmente devido ao balanceamento da árvore AVL.

**Teste de Busca**

Foi realizada a busca pelo valor **25** em ambas as árvores. A contagem representa quantas vezes a função de busca foi acionada (número de comparações feitas até encontrar o elemento).

| **Árvore** | **Comparações** |
| --- | --- |
| **ABB** | 5 |
| **AVL** | 4 |

**Observação:**

* A AVL realizou menos comparações, demonstrando maior eficiência, especialmente em casos onde a ABB fica desbalanceada.

**Conclusões**

* A **Árvore AVL** mantém melhor desempenho de busca e inserção devido ao balanceamento automático.
* A **ABB** é mais simples de implementar, mas pode ter desempenho ruim quando inserções seguem uma ordem crescente ou decrescente.
* As diferenças nos percursos pré-ordem e pós-ordem ilustram como o balanceamento altera a estrutura da árvore.
* O balanceamento da AVL garante sempre complexidade O(log n), tornando-a mais eficiente para conjuntos de dados maiores e operações de busca frequentes.