Universidade Presbiteriana Mackenzie

ESTRUTURA DE DADOS II

Árvore AVL – parte 2/2
Atividade (máx. três alunos)

Objetivo

Implementar uma árvore AVL com operações de inserção e remoção, em Java.

Instruções

- A atividade deve ser resolvida usando a linguagem Java.
- A solução não deve usar as estruturas de dados oferecidas pela linguagem Java (projetos que usarem tais estruturas serão desconsiderados – zero).
- Inclua a identificação do grupo (nomes completo e TIA de cada integrante) no início de cada arquivo de código, como comentário.
- Inclua todas as referências (livros, artigos, sites, vídeos, entre outros) consultadas para solucionar a atividade, como comentário no arquivo `.java` que contém a `main()`.

Enunciado

Agora que você já implementou o cálculo do fator de balanceamento para cada nó de uma árvore AVL e as operações de rotação na classe que representa uma árvore AVL, o foco dessa segunda parte da implementação da árvore AVL é adicionar as operações de inserção e remoção de nós com o balanceamento da árvore.

1. Implemente a operação de inserção de nós em uma árvore AVL (ex. método `insert()` da classe `AVL`).

2. Implemente a operação de remoção de nós em uma árvore AVL (ex. método `remove()` da classe `AVL`).

3. Para testar a sua implementação e mostrar que a implementação está funcionando, na `main`, construa as seguintes árvores de teste (uma árvore por item):

a) Inserir, nessa ordem, os nós com chaves 1, 2 e 3.
b) Inserir, nessa ordem, os nós com chaves 3, 2 e 1.
c) Inserir, nessa ordem, os nós com chaves 3, 1 e 2.
d) Inserir, nessa ordem, os nós com chaves 1, 3 e 2.
e) Inserir, nessa ordem, os nós com chaves 5, 4, 3, 1, 2, 6, 7, 9 e 8.
f) Remover o nó 4 da árvore do item (e).
g) Remover o nó 5 da árvore do item (f).
h) Remover o nó 1 da árvore do item (g).

Para cada item acima (a-h), após a construção da árvore, seu projeto deve exibir os dados atualizados de todos os nós (pelo menos quem é o nó pai, nó filho esquerdo, nó filho direito e o fator de balanceamento).

Universidade Presbiteriana Mackenzie

ESTRUTURA DE DADOS II

Entrega

Compacte o código-fonte (somente arquivos *.java) no **formato zip**.

Atenção: O arquivo zip não deve conter arquivos intermediários e/ou pastas geradas pelo compilador/IDE (ex. arquivos *.class, etc.).

Prazo de entrega: via link do Moodle até 22/10/2023 23:59.

Critérios de avaliação

A nota da atividade é calculada de acordo com os critérios da tabela a seguir.

ITEM AVALIADO	PONTUAÇÃO MÁXIMA
1. Implementação da operação de inserção de nós em uma árvore AVL + balanceamento da árvore.	2,0
2. Implementação da operação de remoção de nós em uma árvore AVL + balanceamento da árvore.	2,0
3. Teste A (1, 2, 3).	0,5
3. Teste B (3, 2, 1).	0,5
3. Teste C (3, 1, 2).	0,5
3. Teste D (1, 3, 2).	0,5
3. Teste E (5, 4, 3, 1, 2, 6, 7, 9, 8).	0,75
3. Teste F (remoção do nó 4 da árvore Teste E).	0,75
3. Teste G (remoção do nó 5 da árvore Teste F).	0,75
3. Teste H (remoção do nó 1 da árvore Teste G).	0,75
Funcionamento geral do programa, de acordo com o enunciado.	1,0

Tabela 1 - Critérios de avaliação.

A tabela a seguir contém critérios de avaliação que podem **reduzir** a nota final da atividade.

ITEM INDESEJÁVEL	REDUÇÃO DE NOTA
O projeto é cópia de outro projeto.	Projeto é zerado
O projeto usa estruturas de dados oferecida pela linguagem Java.	Projeto é zerado
Há erros de compilação e/ou o programa trava durante a execução[1].	-1,0
Não há identificação do grupo. Não há indicação de referências. Arquivos enviados em formatos incorretos. Arquivos e/ou pastas intermediárias que são criadas no processo de compilação ou pela IDE foram enviadas junto com o código-fonte.	-1,0

Tabela 2 - Critérios de avaliação (redução de nota).

O código-fonte será compilado com o compilador javac (17.0.8) na plataforma Windows da seguinte forma:

```
> javac *.java
```

O código compilado será executado com java (17.0.8) na plataforma Windows da seguinte forma (`<Classe>` deve ser substituído pelo nome da classe que contém o método `public static void main(String[] args)`):

```
> java <Classe>
```

[1] Sobre erros de compilação: considere apenas erros. Não há problema se o projeto tiver *warnings* (embora *warnings* podem avisar sobre possíveis travamentos em tempo de execução, como loop infinito, divisão por zero, etc.).