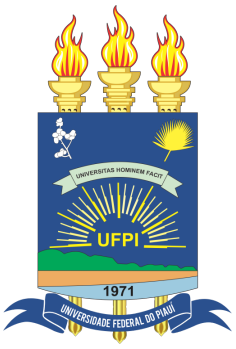

Universidade Federal do Piauí
Alunos: Jhoisnáyra Vitória Rodrigues de Almeida
Pedro Gonçalves Neto.
Curso: Ciência da Computação.

Relatório de Estrutura de Dados:
Árvore binária de busca.

Teresina, 14 de novembro de 2019

.

1. Introdução

Árvores binárias são conjuntos finitos de elementos, podendo o conjunto estar vazio ou composto de 3 subconjuntos, a raiz da árvore, a subárvore da esquerda e a subárvore da direita.

Uma árvore binária com n folhas obrigatoriamente possui *2n - 1* nós, um nó sem filho é denominado folha. A raiz de uma árvore é o único nó que não é filho de outro, ou seja, se a raiz não existir, a árvore está vazia. A profundidade de um nó é definida pela quantidade de links da raiz ao nó, a altura é a profundidade máxima dos nós (sendo no máximo N-1 e no mínimo log n), e o comprimento interno é a soma de todas as profundidades da árvore.

Neste relatório, pretende-se realizar a implementação de uma árvore binária com campo *depth*, que armazene a profundidade, e as funcionalidades de mostrar número de nós, altura, comprimento interno, impressão dos valores (em-ordem, pós-ordem, pré-ordem) e método *setDepthField()* que defina a profundidade de todos os nós do conjunto.

2. Desenvolvimento

Criou-se uma interface GUI para escolha e leitura do arquivo e em seguida realizar as operações descritas pelo professor. Como exemplo para demonstração utilizou-se a árvore binária apresentada no livro *Algorithms*, mostrada na imagem 1, e em seguida realizou-se as operações.

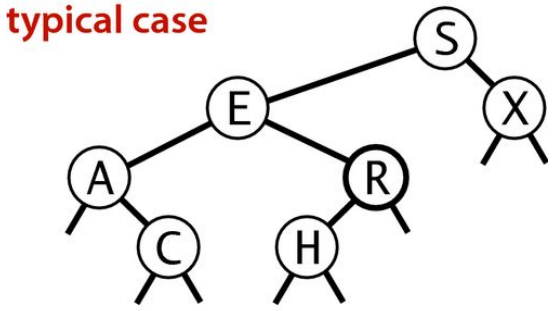

Imagem 1 - Árvore binária em caso típico.

Em seguida realizou-se as operação de verificação de tamanho e altura, que resultam respectivamente em 7 e 3, como mostra a imagem 2.

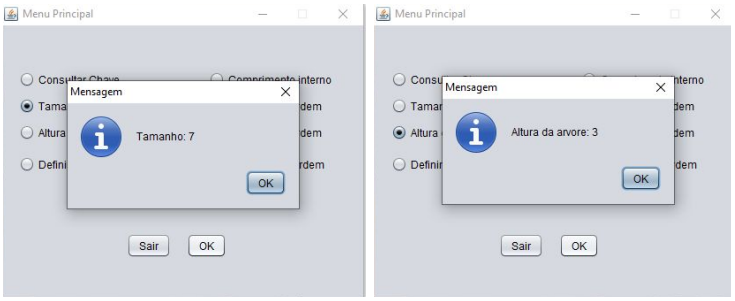

Imagem 2 -Tamanho e altura da árvore binária.

Após isso define-se as profundidades para que seja possível verificar o comprimento interno (Imagem 3) que é igual a 12.

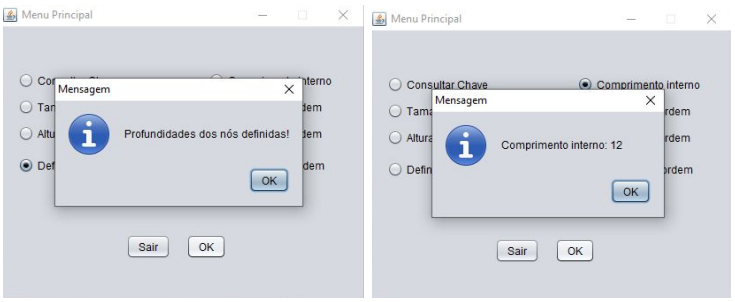

Imagem 3 -Definição das profundidades da árvore binária e verificação do comprimento interno.

Por fim, imprime-se o arquivo em ordem (Imagem 4), entretanto também é possível imprimir em pós ordem e pré ordem.

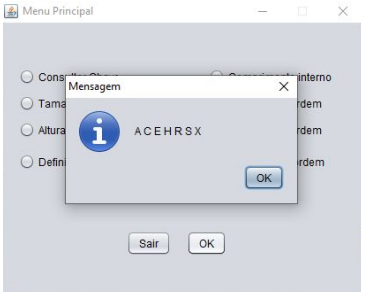

Imagem 4 -Impressão em ordem da árvore binária.

3. Considerações Finais

Conclui-se, portanto que a interface GUI implementada atingiu os resultados esperados, sendo capaz de receber arquivos txt de palavras ou letras e ordená-las de forma correta em uma árvore binária de busca. Caso a árvore se mantenha balanceada, isto é, que o tamanho da subárvore esquerda não ultrapasse o tamanho da subárvore da direita em 1, e vice-versa, atinge-se a chamada performance otimizada ou ótima.

4. Referências:

Princeton (2019). *Java Algorithms and Clients*. [online] Disponível em: https://algs4.cs.princeton.edu/code/ [Acessado em 3 de outubro de 2019].

LAUREANO, Marcos. **ESTRUTURA DE DADOS COM ALGORITMOS EC**. Brasport, 2008.

GRANATYR, Jones et al. Árvores binárias (v. 2). **Almanaque para popularização de ciência da computação. Série 5, Estrutura de dados**, 2016.