TRABALHO 3 - ALGORITMOS E ESTRUTURA DE DADOS I

Gabriela Biasi, Luiz Henrique Borges Mosmann, Nathalia Rodrigues

04 de Dezembro de 2018

1 Introdução

O trabalho a seguir tem como objetivo desenvolver uma solução para gerenciar uma árvore de palavras. Para isso, foi implementado uma árvore genérica usando seus principais métodos. Após deu-se inicio a implementação da interface gráfica e por fim a classe Main que realiza as operações de cada método. Como resultado final, podemos observar que para cada carácter digitado, o programa lista uma serie de nomes ordenados alfabeticamente. Caso seja informado a palavra inteira, o programa localiza e retorna a mesma.

2 Desenvolvimento

O código implementado foi dividido em quatro classes, sendo elas: Palavra, Tela, Arquivo e a própria classe Main. A classe Palavra instancia uma palavra e seus métodos (Get/Set). A classe Tela representa a interface gráfica do projeto, contendo um campo de pesquisa, um botão de busca e um campo onde serão exibidos os resultados. A classe Arquivo contém os principais métodos da estrutura de dados Árvore, como por exemplo: positionsPreAux(), subTree(), getRoot() e addNode(). Além disso é responsável pela leitura do arquivo que contém os nomes para a busca. Por fim a classe Main realiza as operações setadas.

```
Leitura do arqurivo() { // O(n)
    le o arquivo "nomes.csv";
      Faz a  separao ;
}
Busca de Palavras(entrada) {
    Se  no  receber nada/null
    return null;

    Nova lista de palavras do tipo ArrayList;

     Node root = primeiro caractere do nome vira raiz;
     Node subRoot = pega a palavra e armazena cada caractere em um nodo
```

```
 Faz o caminhamento na arvore e verifica se existe significado da
     palavra, se existir retorna a palavra.
}

PositionsPreAux() {
    Se n for nulo
        returna listaDePalavras;


    Se os filhos de n forem diferente de nulos{
        Por i igual a zero indo at a quantidade de filhos de n{
            Chama o mtodo novamente, passando por parmetro o filho da
                posio i da lista de filhos, a palavra incompleta
                concatenado com o elemento de n, as iniciais e tambm
                a lista de palavras.

        }
    }
    Retorna a lista de palavras.
}

Node subTree()

   Se raiz for nula
    retorna raiz.

   Se a pos for igual ao tamanho da entrada
    retorna raiz.

   Se os filhos da raiz nao forem nulo
        Por i comeando em 0 e indo at a quantidade de filhos da raiz{
            Se o elemento contido em filhos for igual ao caractere da
                palavra de entrada na posio pos.{
               raiz = chama o mtodo subTree, passando por parmetro o
                    elemento i da lista sons, a palavra de entrada e
                    a prxima posio. Aps isso, break.


void addNode(){
    Node root = o primeiro caractere da palavra vira raiz.
    Chama o mtodo addNodeAux, passando por parmetro a palavra em
        letras maisculas, o sig, 1, a raiz e "".
}

void addNodeAux(){

    Se pos for igual ao tamanho de palavra
        significado de n recebe sig
    char letra = recebe a letra pertencente a palavra na posio pos.
```

```
    Node aux = recebe nulo.

    Por i comeando em 0 e indo at a quantidade de filhos de n{
        Se a letra for igual ao elemento salvo na lista de filhos.{
            aux = filhos na  posio  i.
            Break;
        }
    }

    Se o auxiliar for nulo{
        aux = novo nodo contendo a letra.
        Adicionar aux na lista de filhos.
    }
    Chama o mtodo addNodeAux, passando por parmetro a palavra, sig,
         pos+1, aux e print concatenado com a letra.
}
```

A imagem abaixo retrata um exemplo da estrutura de dados utilizada no desenvolvimento de nosso projeto.

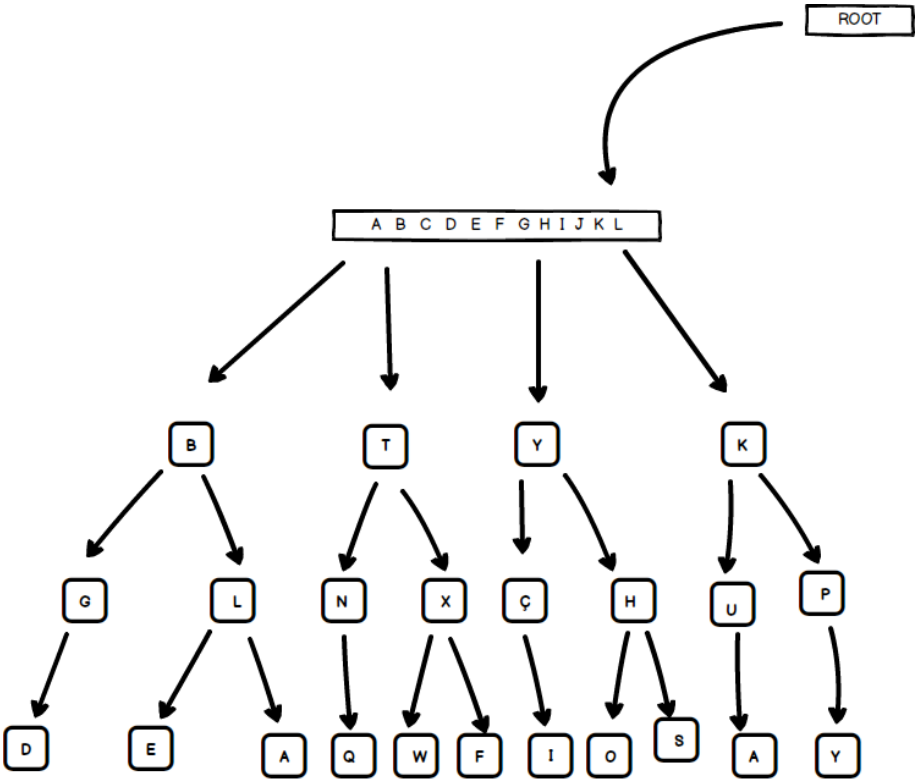

Figura 1: Exemplo de Estrutura de Dados Utilizada

3 Resultados Obtidos

As imagens abaixo apresentam os resultados de cada busca por nome ou carácter realizada. Como podemos observar, a busca só retorna o nome correto, quando o mesmo é especificado por completo, caso contrario retornará todos os nomes que possuem como prefixo a palavra digitada até então. Além disso, para os principais métodos implementados, addNodeAux(), addNode(), e buscaDePalavras(), classificamo-os como O(n). Pois o buscaDePalavras() depende da entrada recebida para percorrer e achar a palavra e o addNode()/addNodeAux() adiciona ate a palavra ficar completa, logo se encaixa em O(n) também.

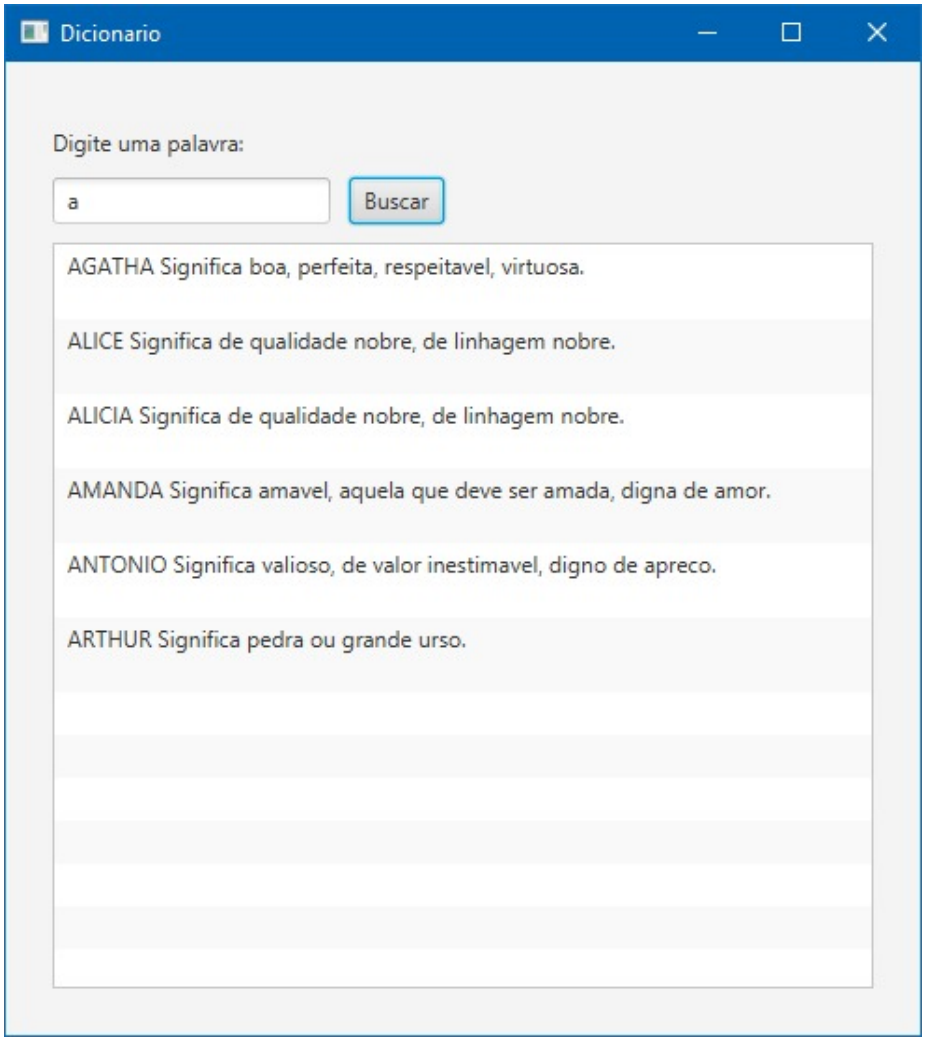

Figura 2: Exemplo de inserção de apenas um caracter

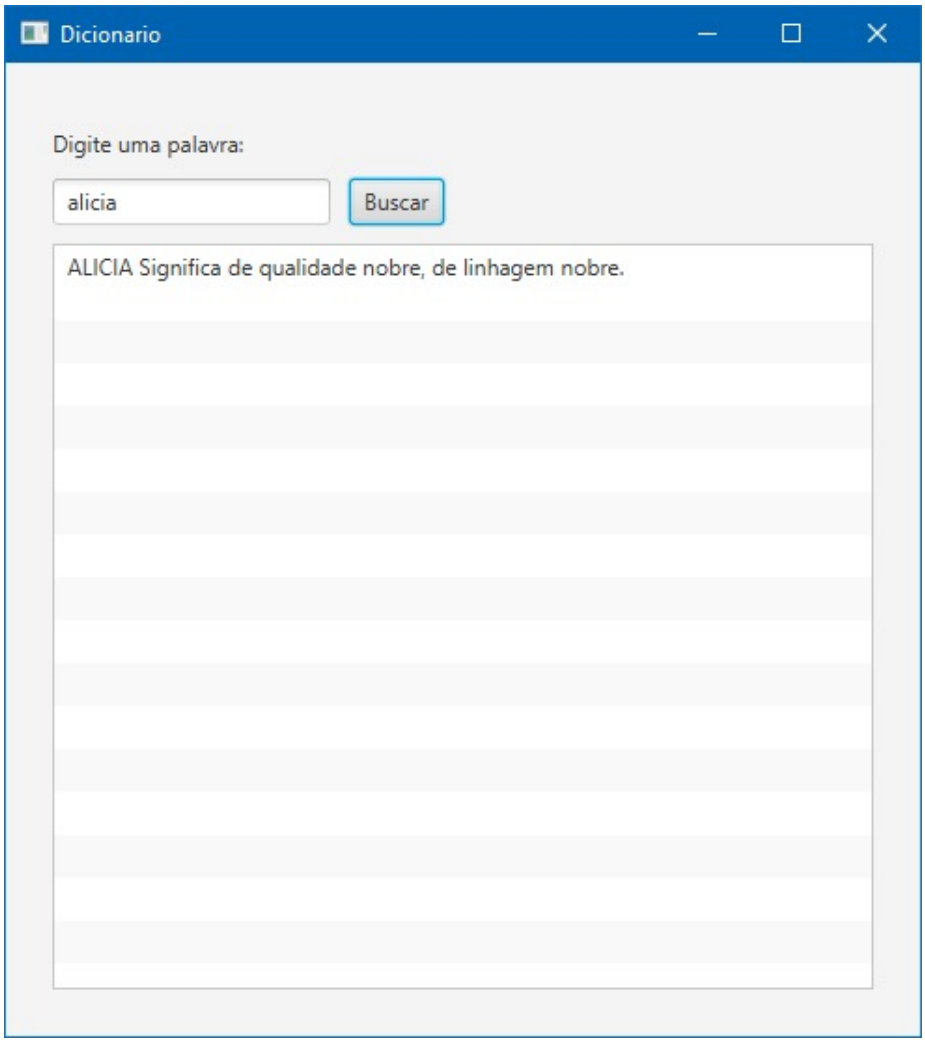

Figura 3: Exemplo de inserção de um nome completo

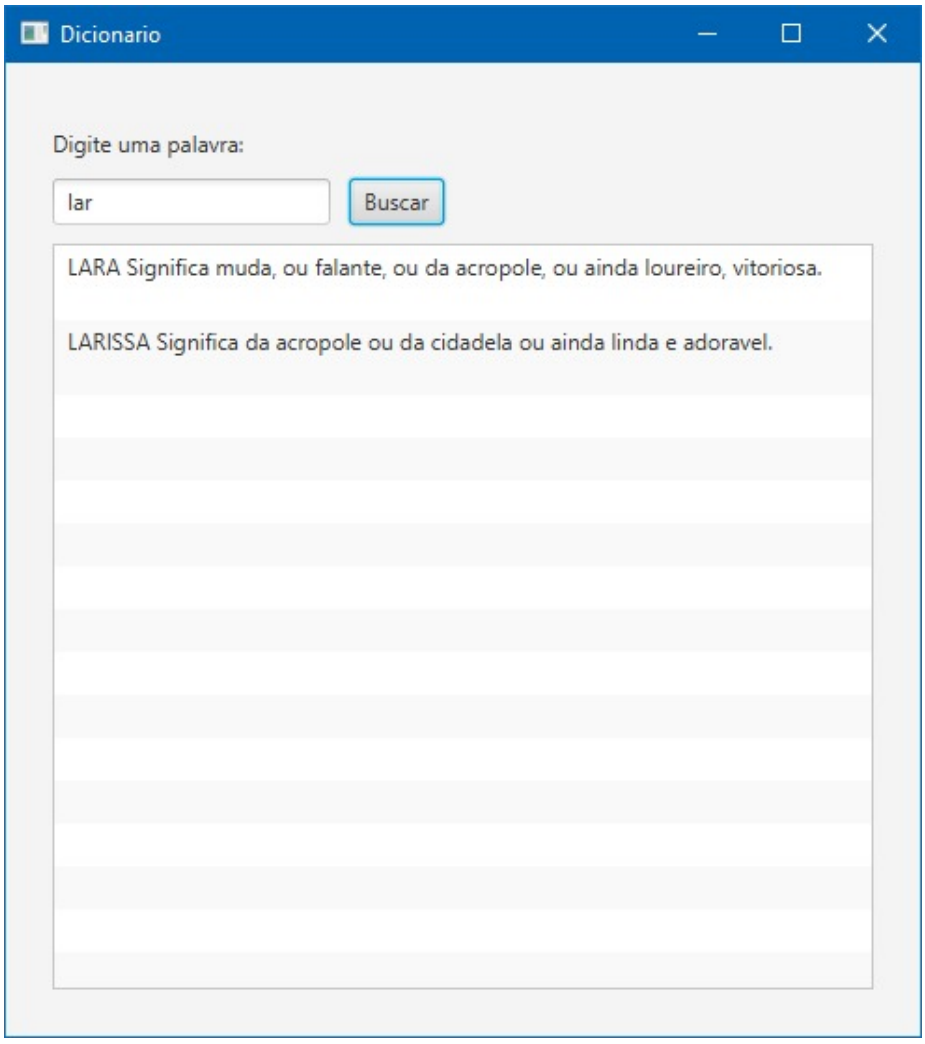

Figura 4: Exemplo de resultado para palavras com caracteres similares

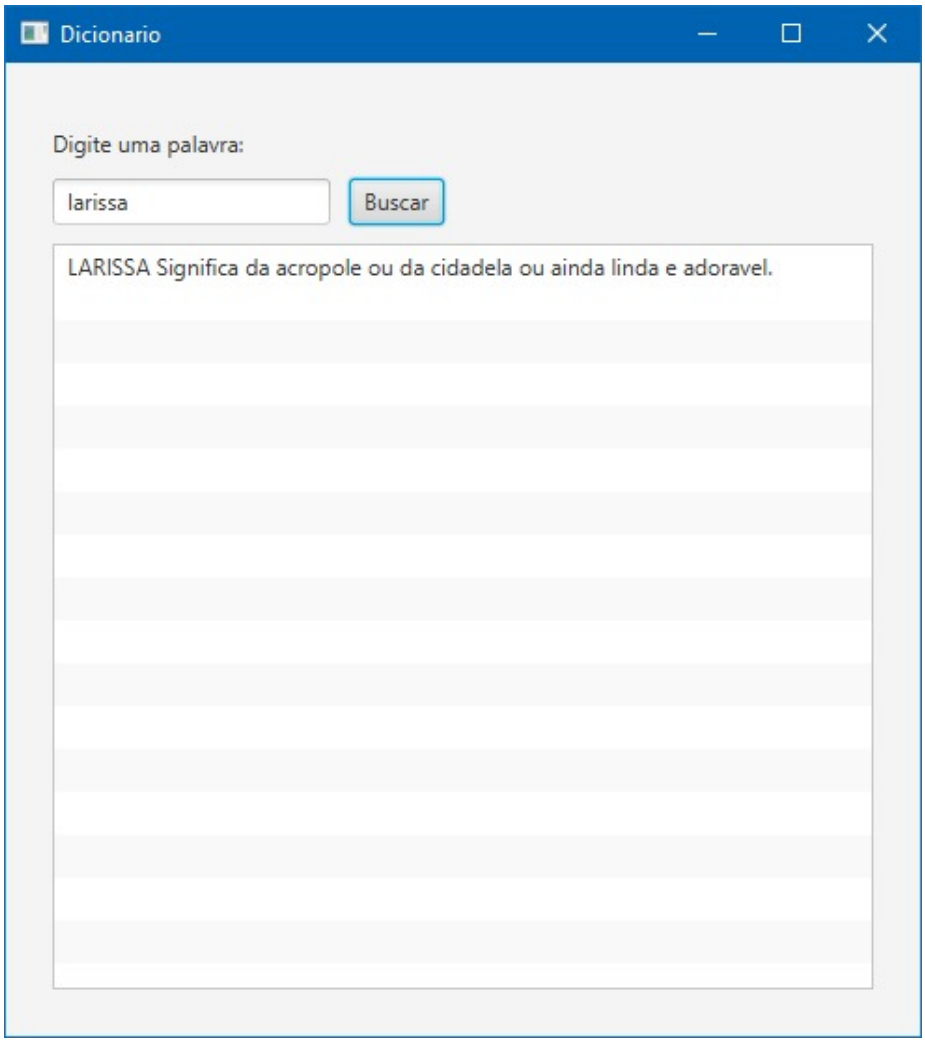

Figura 5: Outro exemplo de inserção de nome completo

4 Conclusão

Pode-se dizer que, uma das principais dificuldades encontradas pelos alunos foi o entendimento do problema trabalhado e quais ferramentas iriamos utilizar para nos auxiliar no desenvolvimento do mesmo.

Por fim, reconhecemos a importância do desenvolvimento deste trabalho no entendimento de cada função que a estrutura árvore utiliza. Além disso, nos fez raciocinar em cada etapa desenvolvida se havia ou não formas mais otimizadas de se fazer o que foi solicitado.