Relatório EDA1

Boogle Game

João Galvão - 37814

Luís Maurício - 37722

Introdução

Este trabalho consiste na resolução de um tabuleiro boogle, o jogo tem como objetivo o mesmo de uma sopa de letras, mas com algumas diferenças, neste jogo pudemos encontrar palavras em qualquer direção apos encontrar uma letra, mas estas não podem ser repetidas apos terem sido encontradas.

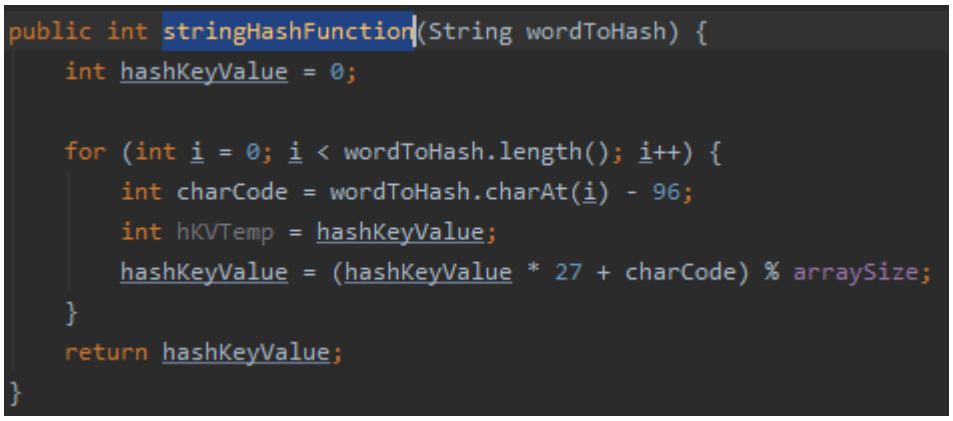
O nosso objetivo é encontrar todas a palavras possíveis num tabuleiro dado, em que as estas são dadas numa lista de todas as palavras do dicionário inglês, em que teremos de usar estruturas de dados para a resolução do problema.

Estruturas Utilizadas

Para a resolução do trabalho utilizamos duas estruras de dados, Hash Tables e Liked Lists.

Utilizamos uma Hash Table para armazenar todas as palavras do dicionário, e tratamos das colisões com Linked Lists, ou seja sempre que houver um colisão esta será introduzida na Linked List associada ao código da Hash Table.

O nosso cálculo de HashCode é calculado com a função `stringHashFunction.`



```
public int stringHashFunction(String wordToHash) {
    int hashKeyValue = 0;

    for (int i = 0; i < wordToHash.length(); i++) {
        int charCode = wordToHash.charAt(i) - 96;
        int hKVTemp = hashKeyValue;
        hashKeyValue = (hashKeyValue * 27 + charCode) % arraySize;
    }
    return hashKeyValue;
}
```

Utilizamos duas Linked Lists uma para tratar das colisões na HasTable como já referido e outra para guardar todas as possíveis palavras da tabela Boogle, que posteriormente serão pesquisadas pelo método `find`, que irá pesquisar pela palavra na Hash Table, em que caso encontre a retorna com as posições referentes a cada letra na tabela.

Classes Utilizadas

Criámos a classe Position que indica as posições de cada letra na matriz de bogle e verifica se esta já foi acedida de maneira a não ser repetida.

Temos a classe Boggle em que o seu objetivo é percorrer a matriz a matriz de bogle e verificar todas as possíveis combinações de letras a verificar na hashtable que contem o dicionário.

A classe Test é classe main onde tudo vai ser executado conforme foi explicito até agora.

A classe Matriz_jogo é a classe que vai retornar a matriz boogle, que irá conter as palavras a pesquisar.

As restantes classes são onde estão implementadas as estruturas de dados explicadas na página anterior.

Conclusão

Neste trabalho verificamos o que a estruturas de dados, mais especificamente as Linked Lists e as Hash Tables, são bastantes uteis e rentaveis em certos casos, vimos que a HashTable é uma boa forma de guardar o dicionário porque a pesquisa de palavras nele é bastante rápida, pois temos acesso direto ao que estamos a procurar excepto em caso de colisão, mas nesta situação as linked lists vem para auxilio fornecendo um melhor rendimento na pesquisa em caso de conflito com hashcode’s identicos.