Laboratório 9: Lista Invertida

Introdução

Muitos sistemas de busca (e.g., Google, Yahoo) usam técnicas de Recuperação de Informação para a indexação e busca rápida em uma base grande de dados (e.g., a Internet). Uma das técnicas mais usadas é a Lista Invertida (ou Índice Invertido).

Em uma lista invertida, as palavras dos documentos (e.g., casa, carro, etc) são indexadas em uma tabela hash e cada item (cada palavra) possui uma lista indicando em quais documentos ela ocorre. Por exemplo, considere os seguintes documentos e os seus conteúdos:

Documento	Conteúdo
`document1.txt`	`Remember, the Force will be with you, always.`
`document2.txt`	`The Force is strong with you.`
`document3.txt`	`The Force is strong with this one.`

Tal base de dados irá gerar a seguinte lista invertida:

Palavra	Documentos
`"force"`	`document1.txt`, `document2.txt`, `document3.txt`
`"always"`	`document1.txt`
`"one"`	`document3.txt`
`"is"`	`document2.txt`, `document3.txt`
`"be"`	`document1.txt`
`"will"`	`document1.txt`
`"you"`	`document1.txt`, `document2.txt`
`"the"`	`document1.txt`, `document2.txt`, `document3.txt`
`"remember"`	`document1.txt`
`"this"`	`document3.txt`
`"strong"`	`document2.txt`, `document3.txt`
`"with"`	`document1.txt`, `document2.txt`, `document3.txt`

Desta forma, para pesquisar todos os documentos que contém a palavra `"strong"`, não precisaríamos pesquisar documento por documento, bastaria pesquisar na tabela hash a palavra (*chave da tabela*) e ver na lista (*valor da tabela*) quais os documentos que a contém (`document2.txt` e `document3.txt`).

Objetivo deste Trabalho

Implementar em Java uma lista invertida, que nada mais é do que uma tabela hash (classe `Hashtable` do Java) em que a *chave* da tabela será uma string (palavra) e em que o *valor* da tabela será uma lista encadeada de strings (lista de documentos). Para a lista encadeada, será usada a classe `LinkedList` do Java.

Sobre as Tabelas Hash

Se você ainda não sabe o que são tabelas hash, para este trabalho basta saber que são estruturas de dados que armazenam pares de chaves/valores, em que os valores são acessados através de suas chaves que, diferentemente dos vetores, não precisam ser números sequenciais, mas qualquer valor ou objeto (incluindo strings). Se você quiser saber os detalhes de como uma tabela hash funciona internamente, dê uma olhada nestes slides de AED2. Apenas por curiosidade, a Tabela Hash (com tamanho 6) do exemplo acima ficará parecido com a figura abaixo.

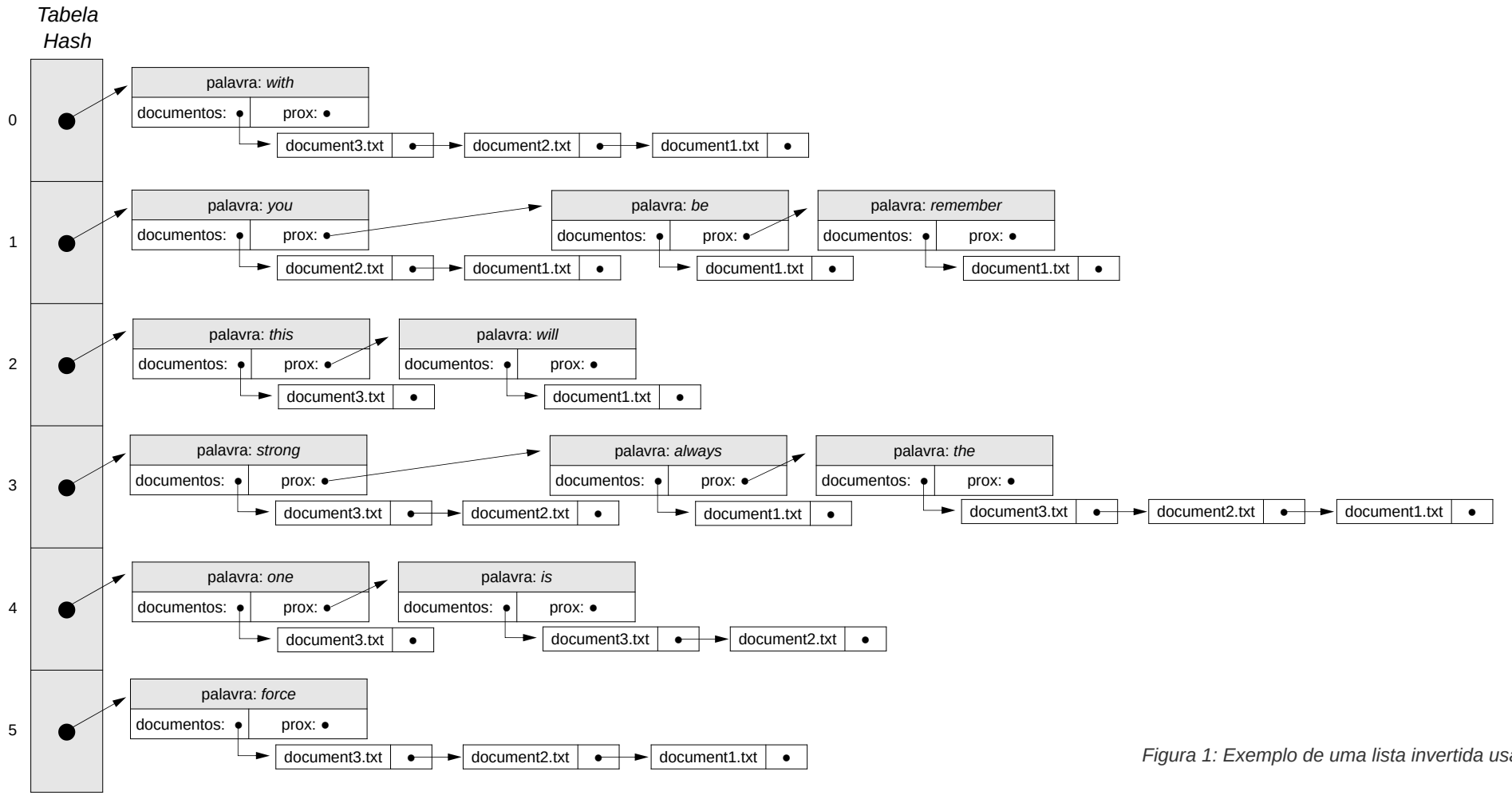
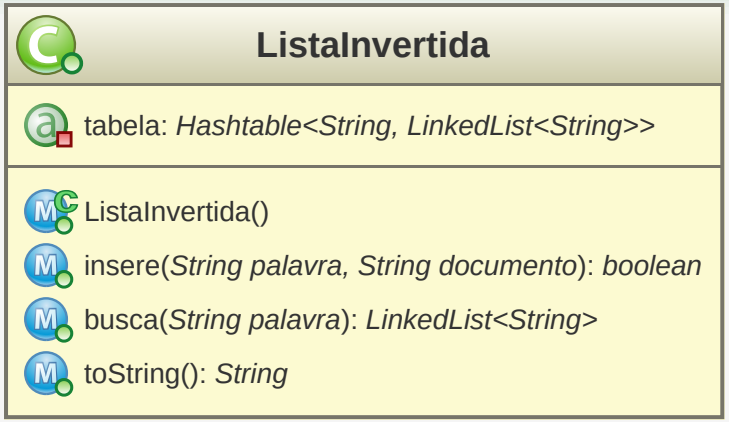


Figura 1: Exemplo de uma lista invertida usando tabelas hash.

Questão 1: Classe `ListaInvertida`



Apesar de ser relativamente complexo, este trabalho é muito pequeno pois utilizaremos as estruturas de dados já implementadas em Java. O trabalho é basicamente implementar esta classe chamada `ListaInvertida`, que combina as estruturas de dados Tabela Hash e Listas Encadeadas da seguinte forma: uma Lista Invertida é uma Tabela Hash cujas chaves são *strings* e cujos valores são *Listas Encadeadas de strings*.

A classe `ListaInvertida` terá apenas um atributo (privado) que será a tabela hash, conforme mostrado no diagrama ao lado.

Seguem algumas observações sobre os métodos:

- `public ListaInvertida()`: no construtor, instancie um novo objeto da classe `Hashtable` para o atributo `tabela`.
- `public boolean insere(String palavra, String documento)`: faça uma busca na tabela hash pela `palavra` usando o método `get` da classe `Hashtable`. Se a palavra for encontrada, a lista de documentos será retornada e, neste caso, verifique se o `documento` já existe na lista (método `contains` da classe `LinkedList`). Se não existir, insira-o e retorne `true`, retornando `false` caso contrário (caso o `documento` já exista na lista de documentos). Se a `palavra` ainda não existe na tabela hash, insira-a (chave) junto com uma nova lista de strings (valor) contendo o `documento` e retornando `true`.
- `public LinkedList<String> busca(String palavra)`: use o método `get` da classe `Hashtable` para buscar a `palavra`.
- `public String toString()`: a classe `Hashtable` já contém um método `toString`, que converte a tabela hash para string. Como os valores da tabela hash (lista de documentos) também possuem o método `toString` (implementado na classe `LinkedList`), estes também serão convertidos automaticamente para strings. Portanto, este método deverá apenas retornar a execução do método `toString` da tabela hash (atributo `tabela`). Exemplo de saída para o exemplo mostrado acima:

```
{force=[document1.txt, document2.txt, document3.txt], always=[document1.txt], one=[document3.txt], is=
[document2.txt, document3.txt], be=[document1.txt], will=[document1.txt], you=[document1.txt, document2.txt],
the=[document1.txt, document2.txt, document3.txt], remember=[document1.txt], this=[document3.txt], strong=
[document2.txt, document3.txt], with=[document1.txt, document2.txt, document3.txt]}
```