



Universidade de Aveiro

Projeto de MPEI  
Estilos de Escrita

Eduardo Coelho - 88867

Ruben Menino - 89185

# Conteúdo

<b>1</b>	<b>Introdução</b>	<b>2</b>
<b>2</b>	<b>Lista de módulos</b>	<b>3</b>
<b>3</b>	<b>Programas a executar</b>	<b>4</b>
3.1	Teste_ContadorEstocastico.java . . . . .	4
3.2	Teste_CountingBloomFilter.java . . . . .	5
3.3	Teste_MinHash.java . . . . .	5
3.4	Aplicação . . . . .	5
<b>4</b>	<b>Manual de utilizador</b>	<b>6</b>
4.1	Como utilizar o programa . . . . .	6
4.1.1	Comparar os excertos escolhidos . . . . .	6
4.1.2	Observações . . . . .	6
<b>5</b>	<b>Conclusão</b>	<b>7</b>

# Capítulo 1

## Introdução

No âmbito do projeto da unidade curricular Métodos Probabilísticos em Engenharia Informática, foi nos proposta a criação de três módulos, incluindo os seus respetivos testes, e uma aplicação à nossa escolha onde temos que usar, pelo menos, alguns dos módulos desenvolvidos. A aplicação escolhida foi testar a similaridade na escrita de diferentes autores. Neste relatório iremos apresentar detalhadamente, todos os módulos presentes neste projeto e explicar como funcionar com a nossa aplicação dos mesmos.

# Capítulo 2

## Lista de módulos

- Contador Estocástico (*ContadorEstocastico.java*)
- Teste do Contador Estocástico (*Teste\_ContadorEstocastico.java*)
- BloomFilter (*CountingBloomFilter.java*)
- Teste do BloomFilter (*Teste\_CountingBloomFilter.java*)
- minHashing (*MinHash.java*)
- Teste do minHashing (*Teste\_MinHash.java*)
- Projeto no terminal (*Projeto.java*)
- Interface gráfica (*MenuProjeto.java*)

# Capítulo 3

## Programas a executar

Este projeto contém 5 ficheiros Java com funções main. Três deles são os respetivos testes de cada módulo(*Teste\_ContadorEstocastico.java*, *Teste\_CountingBloomFilter.java* e *Teste\_MinHash.java*), o outro é o *Projeto.java* que contém a aplicação criada por nós, e o último é a respetiva interface gráfica do nosso projeto (*MenuProjeto.java*).

### 3.1 Teste\_ContadorEstocastico.java

Neste programa criamos um ContadorEstocastico atribuindo para este teste os seguintes argumentos: contador até 1000000 e probabilidade de 50%. O teste consiste em incrementar este contador estocástico a quantidade de vezes definida (neste caso, 10000000), e no fim multiplicar o resultado obtido por 2 (devido a termos definido a probabilidade a 50%) e verificar se realmente o valor obtido é aproximado do valor que definimos para o contador.

## 3.2 Teste\_CountingBloomFilter.java

O teste deste programa foi baseado num dos exercícios presentes no guião de MPEI que aborda o tema dos BloomFilters. Criamos um grupo de países que inserimos no bloomfilter, e depois criamos um outro conjunto de países para testar a sua presença no bloomfilter. Para testar as propriedades adicionais relativas a ser um CountingBloomFilter em vez de apenas BloomFilter, testamos também os dois métodos criados, a `counting()`, onde verificamos quantas vezes foi inserida uma palavra igual, e a `remove()`, que nos permite remover um membro adicionado previamente.

## 3.3 Teste\_MinHash.java

Para testar o funcionamento do minhashing, criamos conjuntos, alguns muito parecidos, e outros muito diferentes, e comparamos os vários conjuntos entre si. Em outro teste, criámos uma String de 5 carateres e comparamos a 100000 outras, criadas aleatoriamente, e contamos quantas tinham acima de 20% de similaridade com a String original. Para ambos os testes usamos a divisão em shingles de 2.

## 3.4 Aplicação

Todos os excertos utilizados são compostos por 3000 carateres (após a remoção dos espaços) para aumentar a realidade dos testes. Poderá executar o programa *MenuProjeto.java* caso pretenda utilizar a interface gráfica criada, ou então o *Projeto.java*, caso prefira usar a aplicação através do terminal.

# Capítulo 4

## Manual de utilizador

### 4.1 Como utilizar o programa

O utilizador poderá usar a interface gráfica desenvolvida por nós executando o ficheiro *MenuProjeto.java*, ou caso tenha encontrado algum inconveniente, tem a opção de utilizar o terminal, *Projeto.java*.

#### 4.1.1 Comparar os excertos escolhidos

Escolha os dois excertos que pretende comparar. Poderá fazer o cálculo da similaridade carregando no botão "Calcular" (ou escolhendo outra checkbox, porém recomendamos utilizar o botão).


#### 4.1.2 Observações

O botão "Resultados" irá abrir o ficheiro *RESULTADOS.txt* que contém todas as combinações possíveis e as suas respetivas similaridades. Caso não consiga utilizar esta funcionalidade, poderá abrir a imagem *RESULTADOSimg.png* presente na pasta do projeto.

## Capítulo 5

## Conclusão

Figura 5.1: Comparação final das similaridades

 RESULTADOS:	— □ ×	
s1 = Ensaio sobre a Cegueira s2 = Intermitências da Morte s3 = Memorial do Convento s4 = Homem Duplicado s5 = Ensaio Sobre a Lucidez	o1 = Canto I o2 = Canto III o3 = Canto X o4 = Os Maias o5 = Cidade e as Serras	
<b>SARAMAGO COM SARAMAGO</b> s1 + s2=0.216 s2 + s3=0.236 s3 + s4=0.210 s4 + s5=0.215 s1 + s3=0.200 s2 + s4=0.232 s3 + s5=0.220 s1 + s4=0.205 s2 + s5=0.234 s1 + s5=0.198		<b>Saramago vs Saramago Média = 0.2166</b>
<b>SARAMAGO COM OUTROS</b> s1 + o1=0.145 s2 + o1=0.167 s3 + o1=0.160 s4 + o1=0.165 s5 + o1=0.164 s1 + o2=0.150 s2 + o2=0.180 s3 + o2=0.158 s4 + o2=0.173 s5 + o2=0.162 s1 + o3=0.140 s2 + o3=0.178 s3 + o3=0.165 s4 + o3=0.148 s5 + o3=0.168 s1 + o4=0.160 s2 + o4=0.184 s3 + o4=0.178 s4 + o4=0.173 s5 + o4=0.185 s1 + o5=0.148 s2 + o5=0.170 s3 + o5=0.173 s4 + o5=0.160 s5 + o5=0.161		<b>Saramago vs Outros Média = 0.1646</b>

No desenvolvimento deste projeto percebemos a importância que testar a similaridade entre conjuntos pode ter em diversas aplicações, desde em base de dados a algoritmos automatizados como as pesquisas nos motores de busca. Em relação à nossa aplicação deste tema, podemos concluir que de facto existe certas tendências linguísticas entre obras do mesmo escritor, o que resulta numa similaridade menor quando se compara obras de escritores diferentes do que quando se faz uma comparação entre duas obras do mesmo.