



Escola de Engenharia

Departamento de Informática

Licenciatura em Engenharia Informática

Universidade do Minho

Projeto de LI3

“Transitórios LEI”

André Santos A60994

Helena Alves A61000

Ricardo Branco A61075

Braga, Maio de 2012

Resumo

Neste relatório está descrita o desenvolvimento do projecto “Transitários LEI” em java desta unidade curricular.

Para além da implementação de inúmeros métodos e classes, o principal foco de atenção esteve nas estruturas escolhidas e, posteriormente, na interface e funcionalidades disponibilizadas ao utilizador.

No entanto, numa primeira etapa foi procurado, escolher quais as melhores estruturas para servirem de base à utilização e desenvolvimento futuro do nosso programa, mas, mais importante, verificar as diferenças e as vantagens de cada uma das estruturas em diferentes operações.

Para ajudar a escolher essas mesmas estruturas, foram feitos testes com diferentes combinações de estruturas, quatro combinações nos clientes e quatro nas localidades.

Foram efectuadas tabelas comparativas e gráficos, de forma a escolher qual a combinação que garante melhor funcionamento do programa.

De seguida, foram efectuados testes de camada de persistência com os respectivos gráficos e tabelas.

Para finalizar, foram criados todos os métodos para todas as funcionalidades necessárias. E consequentemente, foi elaborada uma interface de rápida utilização e, que permite ao utilizador uma boa usabilidade de todas as funcionalidades.

Índice

Resumo.....	2
--------------------	----------

1 Introdução.....	7
--------------------------	----------

2 Testes.....	8
----------------------	----------

2.1 .. Máquina de teste	8
-------------------------------	---

2.2 .. Programas em execução	8
------------------------------------	---

2.3 .. Condições de Teste	8
---------------------------------	---

3 Comparação de Estruturas.....	10
--	-----------

3.1 .. Clientes : Teste 1 vs Teste 2	10
--	----

Teste 1: Estrutura baseada em ArrayList, para os dois critérios: nome e NIF;	10
---	----

Teste 2: Estrutura baseada em ArrayList por nome e uma estrutura auxiliar em LinkedList por NIF;	10
---	----

3.1.1 Inserção de Clientes	10
----------------------------------	----

3.1.2 Procurar Clientes	11
-------------------------------	----

3.1.3 Percorrer e Imprimir	12
----------------------------------	----

3.1.4 Ler ficheiro	13
--------------------------	----

3.2 .. Clientes: Teste 3 vs Teste 4	15
---	----

3.2.1 Inserção de Clientes	15
----------------------------------	----

3.2.2 Procurar Clientes	16
-------------------------------	----

3.2.3 Percorrer e Imprimir	17
----------------------------------	----

3.2.4 Ler Ficheiro	18
--------------------------	----

3.3.. Localidades: Teste 1 vs Teste 2.....	19
--	----

3.3.1 Procurar Localidades	19
----------------------------------	----

3.3.2 Inserir Localidade	20
--------------------------------	----

3.3.3 Inserir Ligação	21
-----------------------------	----

3.3.4 Percorrer e Imprimir	22
----------------------------------	----

3.3.5 Ler ficheiro	22
--------------------------	----

3.4.. Localidades: Teste 3 vs Teste 4.....	23
--	----

3.4.1 Procurar Localidades	24
----------------------------------	----

3.4.2 Inserir Localidades	25
---------------------------------	----

3.4.3 Inserir Ligação	26
-----------------------------	----

3.4.4 Ler ficheiro	27
--------------------------	----

3.4.5 Percorrer e Imprimir	28
----------------------------------	----

3.5.. Leitura e escrita	29
-------------------------------	----

3.5.1 Leitura de Localidades	29
------------------------------------	----

3.5.2 Escrita de Localidades	30
------------------------------------	----

3.5.3 Leitura de Clientes	31
---------------------------------	----

3.5.4 Escrita de Clientes	32
---------------------------------	----

4 Estrutura final.....	34
-------------------------------	-----------

5 Interface	35
--------------------------	-----------

6 Conclusão	38
--------------------------	-----------

7 Informação do grupo39

7.1 .. Fotos 40

7.2 .. Informação 41

Índice de Tabelas

Tabela 1 – Inserção Clientes Teste 1.....	10
Tabela 2 – Inserção Clientes Teste 2.....	10
Tabela 3 – Procura Clientes por Nome Teste 1.....	11
Tabela 4 – Procura Clientes por NIF Teste 1	11
Tabela 5 – Procura Clientes por NIF Teste 2	11
Tabela 6 – Procura Clientes por Nome Teste 2.....	12
Tabela 7- Percorrer e Imprimir Cliente Teste 1.....	13
Tabela 8 – Percorrer e Imprimir Cliente Teste 2.....	13
Tabela 9 – Ler ficheiro Teste 1	14
Tabela 10 – Ler ficheiro Teste 2.....	14
Tabela 11 – Inserção Clientes Teste 3.....	15
Tabela 12 – Inserção Clientes Teste 4.....	15
Tabela 13 – Procura Clientes por NIF Teste 3	16
Tabela 14 – Procura Clientes por Nome Teste 3.....	16
Tabela 15 – Procura Clientes por NIF Teste 4	16
Tabela 16 – Procurar Clientes por Nome Teste 4	16
Tabela 17 – Percorrer e Imprimir Cliente Teste 3.....	17
Tabela 18 – Percorrer e Imprimir Teste 4	17
Tabela 19 – Ler ficheiro Teste 3.....	18
Tabela 20 – Ler ficheiro Teste 4.....	18
Tabela 21 – Procura Localidades Teste 1	19
Tabela 22 – Procura Localidades Teste 2.....	19
Tabela 23 – Inserir Localidade Teste 1.....	20
Tabela 24 – Inserir Localidade Teste 2.....	20
Tabela 25 – Inserção Ligação Teste 1.....	21
Tabela 26 – Inserção ligação Teste 2	21

Tabela 27 – Percorrer e Imprimir Localidade Teste 1	22
Tabela 28 – Percorre e Imprimir Localidade Teste 2	22
Tabela 29 – Ler ficheiro Teste 1	23
Tabela 30 – Ler ficheiro Teste 2	23
Tabela 31 – Procura Localidades Teste 3	24
Tabela 32 – Procura Localidades Teste 4	24
Tabela 33 – Inserção Localidades Teste 3	25
Tabela 34 – Inserção Localidades Teste 4	25
Tabela 35 – Inserção Ligação Teste 3.....	26
Tabela 36 – Inserção Ligação Teste 4.....	26
Tabela 37 –Ler ficheiroTeste 3	27
Tabela 38 – Ler ficheiro Teste 4	27
Tabela 39 – Percorrer e Imprimir Localidade Teste 3	28
Tabela 40 – Percorrer e Imprimir Localidades Teste 4.....	28
Tabela 41 – Leitura Localidades Teste 1.....	29
Tabela 42 – Leitura de Localidades Teste 2.....	30
Tabela 43 – Escrita Localidades Teste 1	30
Tabela 44 – Escrita Localidades Teste 2	31
Tabela 45 – Leitura de clientes Teste 1.....	31
Tabela 46 – Leitura clientes Teste 2.....	32
Tabela 47 – Escrita Clientes Teste 1	32
Tabela 48 – Escrita Clientes Teste 2	33

Índice Ilustrativo

Ilustração 1 – Comparação Inserção Clientes Teste 1 vs Teste 2	11
Ilustração 2 – Comparação Procura Clientes Teste 1 vs Teste 2	12
Ilustração 3 – Comparação Percorrer e Imprimir Teste 1vs Teste 2	13
Ilustração 4 – Comaparação ler ficheiro Teste 1 vs Teste 2	14
Ilustração 5- Comparação Inserção Clientes Teste 3 vs Teste 4	15
Ilustração 6 – Comparação Procurar Clientes Teste 3 vs Teste 4.....	17
Ilustração 7 – Comparação Percorrer e Imprimir Teste 3 vs Teste 4.....	17
Ilustração 8 – Comparação ler ficheiro Teste 3 vs Teste 4	18
Ilustração 9 – comparação Procurar Localidades Teste 1 vs Teste 2.....	19
Ilustração 10 – Comparação Inserir Localidade Teste 1 vs Teste 2.....	20
Ilustração 11 – Comparação Inserir Ligação Teste 1 vs Teste 2	21
Ilustração 12 – Comparação Percorrer e Imprimir Teste 1 vs Teste 2.....	22
Ilustração 13 – Comparação ler ficheiro Teste 1 vs Teste 2	23

Ilustração 14 – Comparação Procurar Localidade Teste 3 vs Teste 4.....	25
Ilustração 15 – Comaparação Inserção Localidades Teste 3 vs Teste 4.....	26
Ilustração 16 – Comparação Inserir Ligação Teste 3 vs Teste 4	27
Ilustração 17 – Comparação Ler ficheiro Teste 3 vs Teste 4.....	28
Ilustração 18 - Comparação Percorrer e Imprimir Teste 3 vs Teste 4.....	29
Ilustração 19 – Comparação Leitura localidades Teste 1 vs Teste 2.....	30
Ilustração 20 – Comparação escrita localidades Teste 1 vs Teste 2.....	31
Ilustração 21 – Comparação Leitura Clienets Teste 1 vs Teste 2	32
Ilustração 22 – Comparação escrita clients Teste 1 vs Teste 2	33
Ilustração 23 – Menu principal – Clientes.....	35
Ilustração 24 – Menu principal - Localidades.....	35
Ilustração 25 – Detalhes cliente	36
Ilustração 26 – Menu Inserir Localidade	36
Ilustração 27 – Menu Calcular distância entre localidades	36
Ilustração 28 – Menu inserir ligação	36
Ilustração 29 – Menu Inserir cliente	37

1 Introdução

No âmbito da cadeira Laboratórios de Informática III, perante o problema apresentado iremos desenvolver em Java um programa que vista ajudar uma empresa de transportes – Transitários LEI – com uma aplicação que permita registar a informação necessária para o seu negócio.

O objectivo é criar um programa que gere as localidades que trabalham com esta empresa, as suas ligações e os clientes. Permitindo, assim, que o utilizador possa interagir de forma fácil com as funcionalidades que a empresa disponibiliza, ou seja, com as funcionalidades que o programa disponibiliza.

Numa primeira etapa, foram desenvolvidas oito versões do nosso programa, cada uma delas com diferentes implementações, principalmente no que se refere às estruturas de dados, com o intuito de verificar qual é a melhor escolha para as estruturas do programa.

O principal objectivo desta fase foi perceber quais as melhores estruturas para efectuar determinadas operações, e, por fim, escolher a estrutura para servir de base para as próximas fases.

Numa segunda etapa, foram feitos testes de comparação entre streams de texto e streams de objecto. Contudo, também foram criados métodos das funcionalidades disponíveis ao utilizador e uma interface, que permite ao utilizador desfrutar de uma boa usabilidade com todas as funcionalidades fornecidas.

2 Testes

2.1 Máquina de teste

O nosso programa foi testado sempre na mesma máquina, que tem como características:

- Asus N61JQ-JX004V;
- Processador: Intel® Core(TM) i7-720QM Quad-Core 1,60GHz; Turbo Boost até 2,80GHz;
- Cache: 6MB Intel® Smart Cache;
- Memória: 6GB DDR3-1066;
- Disco rígido: 500GB SATA (5400rpm);
- Sistema Operativo: Windows 8 Consumer Preview;

É importante salientar, que no momento dos testes, a máquina encontrava-se no plano de energia de alto desempenho.

2.2 Programas em execução

No decorrer dos testes do programa, os programas do computador em execução eram os seguintes:

- Programas Base do Windows;
- Microsoft Excel;
- Dropbox;

No decorrer dos testes do programa preocupamo-nos em ter o menor número possível de programas em execução.

2.3 Condições de Teste

Antes de ser registado os tempos dos vários testes, foram realizar testes-ensaio com o objectivo de aquecer a máquina, de forma a obter resultados melhores e mais coerentes.

No intuito de conseguir medir tempos relativamente baixos e precisos foram feitas várias repetições das procuras, inserções,... . Estas repetições foram atribuídas de forma diferente, nas estruturas

com funcionalidades mais rápidas foram feitas mais repetições e vice-versa.

De igual modo, também foram realizados vários ensaios para cada um dos patamares de número de elementos, de forma a obter uma média aceitável nos diferentes ensaios.

Nas diferentes comparações efectuadas entre as estruturas, foi tido em conta que algoritmos das mesmas estruturas fossem semelhantes, de forma a comparar as estruturas e não os algoritmos.

3 Comparação de Estruturas

3.1 Clientes : Teste 1 vs Teste 2

Teste 1: Estrutura baseada em ArrayList, para os dois critérios: nome e NIF;

Teste 2: Estrutura baseada em ArrayList por nome e uma estrutura auxiliar em LinkedList por NIF;

3.1.1 Inserção de Clientes

Na inserção de Clientes são inseridos 1000 clientes com NIF aleatório, dentro dos valores normais, em ambas as estruturas. De seguida, estes valores são divididos por 1000, para ser determinado o tempo médio de inserir uma unidade.

Inserir Cliente (1000)				
	5000	10000	15000	18000
	µs	µs	µs	µs
Média	20,11667	22,92333	25,74333	26,57667
Mediana	20,3	23,4	25	25
Moda	20,3	25	24,9	24,9
Desvio-padrão	4,97768	3,865023	3,606415	3,553792

Tabela 1 – Inserção Clientes Teste 1

Inserir Cliente (1000)				
	5000	10000	15000	18000
	µs	µs	µs	µs
Média	26,88667	33,06333	38,58667	39,36
Mediana	28,05	32,7	32,8	34,3
Moda	28,1	31,2	31,2	32,8
Desvio-padrão	4,835838	4,844868	13,65314	12,67772

Tabela 2 – Inserção Clientes Teste 2

Pela observação das tabelas, verifica-se que o tempo de inserção no Teste 1 é inferior ao Teste 2. Apesar de a inserção de clientes em LinkedList não precisar de deslocar elementos, no ArrayList, o facto de ter acesso directo às posições, compensa este facto. Então, pode-se concluir que nas procuras numa estrutura baseada em ArrayList é muito melhor escolha que uma em LinkedList.

Ilustração 1 – Comparação Inserção Clientes Teste 1 vs Teste 2

3.1.2 Procurar Clientes

Em ambas as procuras de clientes, procura por NIF e procura por nome, faz-se a procura do NIF/Nome de um cliente aleatório que já esteja inserido. Neste caso, são procurados 1000 clientes e, de seguida, os resultados obtidos são divididos por 1000 para ser determinado o tempo médio de procurar uma unidade.

Procurar Cliente por NIF (1000)				
	5000	10000	15000	18000
	µs	µs	µs	µs
Média	2,55	5,51	6,5466667	7,17
Mediana	3,1	6,2	6,3	7,8
Moda	3,1	4,7	6,2	7,8
Desvio-padrão	0,7646275	0,7992885	1,5368062	0,7320967

Procurar Cliente por Nome (1000)				
	5000	10000	15000	18000
	µs	µs	µs	µs
Média	7,743333	12,68667	16,01	19,02667
Mediana	7,8	10,15	14,1	18,7
Moda	7,8	9,4	14	18,7
Desvio-padrão	0,956172	4,562042	2,934356	2,371081

Tabela 3 – Procura Clientes por Nome Teste 1

Procurar Cliente por NIF (1000)				
	5000	10000	15000	18000
	µs	µs	µs	µs
Média	2,183333	4,833333	8,106667	8,94
Mediana	1,6	4,7	7,8	9,3
Moda	1,6	4,7	7,8	9,4
Desvio-padrão	0,960992	0,488017	1,787401	1,083926

Tabela 5 – Procura Clientes por NIF Teste 2

Procurar Cliente por Nome (1000)				
	5000	10000	15000	18000
	μs	μs	μs	μs
Média	5,946667	12,58333	16,06667	20,12333
Mediana	6,2	12,5	15,6	18,7
Moda	4,7	10,9	15,6	26,5
Desvio-padrão	1,104453	1,872502	2,73029	4,019666

Tabela 6 – Procura Clientes por Nome Teste 2

Como se verifica no gráfico a baixo, o tempo de procura de clientes no Teste 1 é inferior ao Teste 2. Na procura por NIF, o Teste 1 é o que apresenta tempos mais rápidos pois beneficia dos acessos directos aos elementos. A semelhança entre os tempos de procura de clientes por nome, em ambos os teste, é perfeitamente normal, pois estes partilham iguais estruturas. A procura por NIF no Teste 2 apresenta os piores tempos porque nesta estrutura é necessário percorrer os elementos.

Ilustração 2 – Comparação Procura Clientes Teste 1 vs Teste 2

3.1.3 Percorrer e Imprimir

Nesta funcionalidade, como os tempos são mais elevados não é necessário fazer mais que uma listagem. Contudo, também são realizados 30 ensaios. Percorre-se lista de clientes, são colocados os dados num buffer e, no final, é tudo imprimido utilizando um `println`, `println` esse do buffer total.

Percorrer e Imprimir				
	5000	10000	15000	18000
	s	s	s	s
Média	21,027	23,59067	26,06033	27,23567
Mediana	19,575	21,99	24,88	26,75
Moda	16,07	33,07	22,93	26,52
Desvio-padrão	4,343206	4,491575	5,804072	3,707304

tabela 7- Percorrer e Imprimir Cliente Teste 1

Percorrer e Imprimir				
	5000	10000	15000	18000
	s	s	s	s
Média	4,929	13,83067	18,084	21,08933
Mediana	4,835	13,96	17,935	18,72
Moda	4,68	14,51	15,28	17,47
Desvio-padrão	0,291268	1,217264	2,423237	4,103639

tabela 8 – Percorrer e Imprimir Cliente Teste 2

Observando o gráfico, verifica-se que o tempo de percorrer e imprimir é superior no Teste 1 do que no Teste 2. Então, conclui-se que se obtém melhores resultados ao percorrer uma LinkedList do que um Arraylist. É, então, uma estrutura mais indicada que o

ArrayList para casos em que se tenha de fazer várias listagens.

Ilustração 3 – Comparação Percorrer e Imprimir Teste 1vs Teste

3.1.4 Ler ficheiro

De igual modo com a funcionalidade anterior, como os tempos são mais elevados, também, só é necessário fazer uma leitura. Contudo, também são realizados 30 ensaios.

Ler ficheiro				
	5000	10000	15000	18000
	s	s	s	s
Média	0,509667	2,048667	4,622333	6,639667
Mediana	0,47	2,03	4,68	6,7
Moda	0,47	2,03	4,68	6,55
Desvio-padrão	0,069306	0,068869	0,076575	0,087828

Tabela 9 – Ler ficheiro Teste 1

Ler ficheiro				
	5000	10000	15000	18000
	s	s	s	s
Média	3,665667	14,78733	31,61333	45,00233
Mediana	3,74	15,75	31,195	44,77
Moda	3,74	15,91	31,04	45,86
Desvio-padrão	0,134438	1,220523	1,294684	1,278185

Tabela 10 – Ler ficheiro Teste 2

Podemos concluir, através das tabelas, que esta funcionalidade é mais rápida no Teste 1. No entanto, visto que, ao ler um ficheiro vão ser adicionados

clientes, estes tempos vão ser quase proporcionais aos tempos das inserções de clientes.

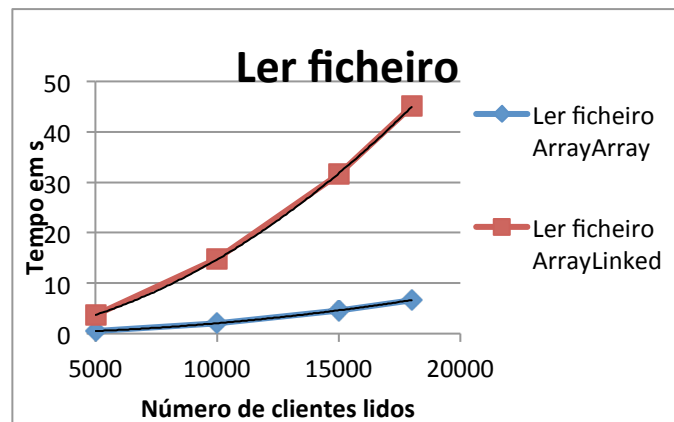


Ilustração 4 – Comparação ler ficheiro Teste 1 vs Teste 2

3.2 Clientes: Teste 3 vs Teste 4

Teste 3: estrutura baseada em HashMap, para utilizadores;

Teste 4: estrutura baseada em TreeMap, para utilizadores;

3.2.1 Inserção de Clientes

Nestes dois testes, como os tempos de inserções e procura são inferiores e, para conseguir registar tempos, houve a necessidade de haver um maior número de inserções/procuras. Sendo assim, neste caso, na inserção de Clientes (Teste 3 e 4) são inseridos 100000 clientes com NIF aleatório, dentro dos valores normais, em ambas as estruturas. De seguida, estes valores são divididos por 100000, para ser determinado o tempo médio de inserir uma unidade

Inserir Cliente (100000)				
	5000	10000	15000	18000
	ns	ns	ns	ns
Média	4,926667	5,113333	5,233333	5,32
Mediana	4,65	4,75	5,45	5,45
Moda	3,2	6,2	6,2	6,2
Desvio-padrão	2,32081	1,70693	2,004035	2,10474

Tabela 11 – Inserção Clientes Teste 3

Inserir Cliente (100000)				
	5000	10000	15000	18000
	ns	ns	ns	ns
Média	7,43	7,44	7,51	7,983333
Mediana	7,75	7,75	7,8	7,8
Moda	7,8	7,8	7,8	7,8
Desvio-padrão	2,71473	2,620819	2,207617	2,541664

Tabela 12 – Inserção Clientes Teste 4

Pela observação das tabelas, verifica-se que o tempo de inserção no Teste 1 é inferior ao Teste 2. No entanto, verifica-se que a inserção de clientes em HashMap é mais rápida que em TreeMap, devido, certamente, aos acessos quase imediatos das posições. Pode-se concluir, também, que ambas são excelentes opções para a inserção de clientes.

3.2.2 Procurar Clientes

Em ambas as procuras de clientes, procura por NIF e procura por nome, faz-se a procura do NIF/Nome de um cliente aleatório que já esteja inserido. Neste caso, são procurados 100000 clientes e, de seguida, os resultados obtidos são divididos por 100000 para ser determinado o tempo médio de procurar uma unidade.

Procurar Cliente por NIF (100000)				
	5000	10000	15000	18000
	µs	µs	µs	µs
Média	4,216667	4,246667	4,576667	4,97
Mediana	3,2	4,6	4,65	4,7
Moda	3,1	4,6	3,1	3,1
Desvio-padrão	1,800208	1,48248	1,553361	2,035402

Tabela 13 – Procura Clientes por NIF Teste 3

Procurar Cliente por Nome (100000)				
	5000	10000	15000	18000
	ns	ns	ns	ns
Média	4,793333	4,973333	5,31	5,593333
Mediana	4,65	4,7	6,2	4,8
Moda	6,2	6,2	6,3	4,7
Desvio-padrão	1,699479	2,056434	2,089028	1,777432

Tabela 14 – Procura Clientes por Nome Teste 3

Procurar Cliente por NIF (100000)				
	5000	10000	15000	18000
	µs	µs	µs	µs
Média	5,733333	6,166667	6,966667	7,236667
Mediana	6,3	6,2	7	7
Moda	6,3	7,8	9,4	6,3
Desvio-padrão	2,494039	1,781966	2,23382	2,466113

Tabela 15 – Procura Clientes por NIF Teste 4

Procurar Cliente por Nome (100000)				
	5000	10000	15000	18000
	ns	ns	ns	ns
Média	7,436667	8,32	9,47	9,966667
Mediana	7,7	7,8	9,4	10,8
Moda	7,8	7,8	9,3	10,9
Desvio-padrão	1,45992	1,900163	1,737836	2,214269

Tabela 16 – Procurar Clientes por Nome Teste 4

Como se verifica no gráfico a baixo, o tempo de procura de clientes no Teste 1 é inferior ao Teste 2. Pode-se concluir que a procura de clientes em HashMap é mais rápida que em TreeMap, devido, certamente, aos acessos quase imediatos as posições. Podemos concluir também que ambas são excelentes opções para a procura de clientes.

Ilustração 6 – Comparação Procurar Clientes Teste 3 vs Teste 4

3.2.3 Percorrer e Imprimir

Nesta funcionalidade, como os tempos são mais elevados não é necessário fazer mais que uma listagem. Contudo, também são realizados 30 ensaios.

Percorrer e Imprimir				
	5000	10000	15000	18000
	s	s	s	s
Média	5,225667	11,075	20,92833	22,041
Mediana	5,15	10,995	20,67	20,9
Moda	5,15	11,54	18,25	19,81
Desvio-padrão	0,176414	0,717273	2,416326	2,85119

Percorrer e Imprimir				
	5000	10000	15000	18000
	s	s	s	s
Média	4,939667	12,214	18,27133	24,90067
Mediana	4,99	11,31	18,325	25,19
Moda	4,99	10,61	15,13	31,82
Desvio-padrão	0,097148	1,595594	2,89754	5,196154

Tabela 18 – Percorrer e Imprimir Teste 4

Observando o gráfico, verifica-se que o tempo de percorrer e imprimir é superior no Teste 1 do que no Teste 2. Perla análise dos gráficos verifica-se que os dois testes apresentam tempos bastante semelhantes, desta vez, ao contrário dos outros testes, verificamos que as estruturas em Map são piores para situações de percorrer e imprimir.

3.2.4 Ler Ficheiro

Nesta funcionalidade, como os tempos são maiores que os restantes, mas continuando com valores pequenos só foi necessário fazer 10 leituras. Contudo, também são realizados 30 ensaios.

Ler ficheiro (10)				
	5000	10000	15000	18000
	ms	ms	ms	ms
Média	8,68	17,47	25,68333	31,09333
Mediana	9,3	17,2	25	31,2
Moda	7,8	17,2	26,5	31,2
Desvio-padrão	0,790635	1,037952	0,995198	0,992014

Tabela 19 – Ler ficheiro Teste 3

Ler ficheiro (10)				
	5000	10000	15000	18000
	ms	ms	ms	ms
Média	12,58333	26,57	40,4	49,65667
Mediana	12,5	26,5	40,6	49,9
Moda	12,5	26,5	40,6	49,9
Desvio-padrão	0,919176	1,327988	1,327893	1,491975

Tabela 20 – Ler ficheiro Teste 4

Podemos concluir, através das tabelas, que esta funcionalidade é mais rápida no Teste 3. No entanto, visto que, ao ler um ficheiro vão ser adicionados clientes, estes tempos vão ser quase proporcionais aos tempos das inserções de clientes.

Ilustração 8 – Comparação ler ficheiro Teste 3 vs Teste 4

3.3 Localidades: Teste 1 vs Teste 2

Teste 1: estrutura baseada em ArrayList, para fazer a gestão das localidades e um ArrayList para as localidades relacionadas;

Teste 2: uma estrutura baseada em ArrayList, para fazer a gestão das localidades e um HashSet para as localidades relacionadas;

3.3.1 Procurar Localidades

Nas procuras de localidades é feito a procura do nome de uma localidade aleatória que já esteja inserida. Neste caso, são procuradas 1000 localidades e, de seguida, os resultados obtidos são divididos por 1000 para ser determinado o tempo médio de procurar uma unidade.

Procurar Localidade (1000)				
	5000	10000	15000	18000
	μs	μs	μs	μs
Média	2,076667	3,896667	6,446667	7,283333
Mediana	1,6	4,6	6,25	7,8
Moda	1,5	4,7	7,8	7,8
Desvio-padrão	0,759166	0,883755	1,129791	1,189344

Tabela 21 – Procura Localidades Teste 1

Procurar Localidade (1000)				
	5000	10000	15000	18000
	μs	μs	μs	μs
Média	2,076667	4,676667	7,486667	8,883333
Mediana	1,6	4,7	7,8	9,3
Moda	1,6	4,7	7,8	7,8
Desvio-padrão	0,744489	0,705976	0,953229	1,005874

Tabela 22 – Procura Localidades Teste 2

Como se verifica no gráfico a baixo, o tempo de procura de clientes no Teste 1 é um pouco inferior ao Teste 2. A semelhança entre os tempos de procura, em ambos os testes, é natural, pois estes partilham iguais estruturas.

3.3.2 Inserir Localidade

Na inserção de localidades são inseridas 1000 localidades com nome aleatório. De seguida, estes valores são divididos por 1000, para ser determinado o tempo médio de inserir uma unidade.

Inserir Localidade (1000)				
	5000	10000	15000	18000
	μs	μs	μs	μs
Média	14,92667	24,59	38,47333	50,9
Mediana	15,6	25	39	47,55
Moda	15,6	23,4	46,8	39
Desvio-padrão	2,541101	3,66835	6,362109	12,95544

abela 23 – Inserir Localidade Teste 1

Inserir Localidade (1000)				
	5000	10000	15000	18000
	μs	μs	μs	μs
Média	4,836667	9,68	14,00333	17,88667
Mediana	4,7	9,4	14,1	17,2
Moda	4,7	9,4	14,1	17,2
Desvio-padrão	0,954295	0,859992	0,977972	1,138279

abela 24 – Inserir Localidade Teste 2

Pela observação das tabelas, verifica-se que o tempo de inserção no Teste 1 é superior ao Teste 2. Apesar de o teste 1 ter apresentado valores superiores ao teste 2, como se está a referir a inserções em iguais estruturas, os resultados deveriam ser mais semelhantes. No entanto, o que se verificou foi, que, efectivamente, no teste 1 obteve-se piores resultados.

Ilustração 10 – Comparação Inserir Localidade Teste 1 vs Teste 2

3.3.3 Inserir Ligação

Na inserção de ligações, é inserida sempre a mesma ligação, em localidades aleatórias já inseridas na lista de localidades. Desta forma, são inseridas 1000 ligações. De seguida, estes valores são divididos por 1000, para ser determinado o tempo médio de inserir uma unidade.

O tempo de inserção de ligações é muito semelhante em ambos os testes. Estes resultados devem-se ao facto de, nos dados teste, o número de ligações de cada localidade não ser significativo. No entanto, num grande número de ligações, em princípio haveria uma maior diferença entre os testes, a beneficiar o teste 2.

Inserir Ligação (1000)				
	5000	10000	15000	18000
	µs	µs	µs	µs
Média	7,703333	15,34	22,98	28,85333
Mediana	7,8	15,6	23,4	28,85
Moda	7,8	15,6	23,4	29,6
Desvio-padrão	0,89961	1,299231	0,997633	1,270822

Tabela 25 – Inserção Ligação Teste 1

Inserir Ligação (1000)				
	5000	10000	15000	18000
	µs	µs	µs	µs
Média	7,95	15,54333	23,19333	29,63333
Mediana	7,8	15,6	23,4	29,6
Moda	7,8	15,6	23,4	29,6
Desvio-padrão	1,112236	0,74217	0,855034	0,917017

Tabela 26 – Inserção ligação Teste 2

Ilustração 11 – Comparação Inserir Ligação Teste 1 vs Teste 2

3.3.4 Percorrer e Imprimir

Nesta funcionalidade, como os tempos são mais elevados não é necessário fazer mais que uma listagem. Contudo, também são realizados 30 ensaios

Percorrer e Imprimir				
	5000	10000	15000	18000
	s	s	s	s
Média	38,66967	43,04833	47,26333	50,75267
Mediana	38,765	43,28	46,95	50
Moda	38,84	44,46	46,95	51,32
Desvio-padrão	0,46917	1,217932	1,503741	2,067786

Tabela 27 – Percorrer e Imprimir Localidade Teste 1

Percorrer e Imprimir				
	5000	10000	15000	18000
	s	s	s	s
Média	39,26733	44,285	50,19	53,327
Mediana	39,385	44,145	49,84	52,875
Moda	37,59	46,64	49,45	54,91
Desvio-padrão	1,343593	1,461732	1,427079	2,005754

Tabela 28 – Percorre e Imprimir Localidade Teste 2

Pela observação do gráfico, pode-se verificar que os tempos são bastante semelhantes, o que era de esperar, pelo facto das localidades estarem guardadas em estruturas semelhantes, e o número de ligações não ser muito significativo.

Ilustração 12 – Comparação Percorrer e Imprimir Teste 1 vs Teste 2

3.3.5 Ler ficheiro

Como os tempos, nesta funcionalidade, são mais elevados, também, só é necessário fazer uma leitura. Contudo, também são realizados 30 ensaios.

Ler ficheiro				
	5000	10000	15000	18000
	s	s	s	s
Média	3,093667	9,993667	24,776	39,99
Mediana	3,12	9,99	25,27	42,505
Moda	3,12	10,14	25,27	42,43
Desvio-padrão	0,100051	0,330209	1,443076	4,668229

tabela 29 – Ler ficheiro Teste 1

Ler ficheiro				
	5000	10000	15000	18000
	s	s	s	s
Média	3,749	14,23133	26,71	42,81867
Mediana	3,74	14,505	27,065	43,285
Moda	3,59	14,35	27,61	44,14
Desvio-padrão	0,222298	0,819364	0,934637	1,451085

tabela 30 – Ler ficheiro Teste 2

Podemos concluir, através das tabelas, que o tempo desta funcionalidade é semelhante em ambos os testes. No entanto, visto que, ao ler um ficheiro vão ser adicionados clientes, estes tempos vão ser quase proporcionais aos tempos das inserções de localidades.

Ilustração 13 – Comparação ler ficheiro Teste 1 vs Teste 2

3.4 Localidades: Teste 3 vs Teste 4

Teste 3: a estrutura baseada em HashMap, para localidades com um HashMap, para as localidades relacionadas;

Teste 4: estrutura baseada em TreeMap, para localidades com um TreeMap, para as localidades relacionadas;

3.4.1 Procurar Localidades

Nestes dois testes, como o tempo de inserções e procuras são inferiores e, para conseguir registar tempos, houve a necessidade de haver um maior número de inserções/procuras. Sendo assim, neste caso, são procuradas 100000 localidades e, de seguida, os resultados obtidos são divididos por 100000 para ser determinado o tempo médio de procurar uma unidade. Nesta procura é realizada a procura do nome de uma localidade aleatória que já esteja inserida.

Procurar Localidade (100000)				
	5000	10000	15000	18000
	ns	ns	ns	ns
Média	6,53	6,7	7,216667	7,386667
Mediana	6,2	6,3	7	7,7
Moda	6,2	6,2	6,2	7,8
Desvio-padrão	1,920695	2,238226	1,964249	1,863207

abela 31 – Procura Localidades Teste 3

Procurar Localidade (100000)				
	5000	10000	15000	18000
	µs	µs	µs	µs
Média	10,71333	11,30333	11,32333	12,07
Mediana	10,9	10,9	11,7	12,4
Moda	9,3	10,9	12,5	11
Desvio-padrão	2,533055	2,101639	2,255797	2,197043

Tabela 32 – Procura Localidades Teste 4

Pela análise do gráfico pode-se verificar que, ao contrário do que se esperava, o teste 4 apresenta melhores resultados que o teste 3, apesar de se estar à espera de melhores resultados do teste 3 em relação ao teste 4. Na realidade, verificou-se que o teste 3 apresentou melhores resultados.

ustração 14 – Comparação Procurar Localidade Teste 3 vs Teste 4

3.4.2 Inserir Localidades

Na inserção de localidades são inseridas 100000 localidades com nome aleatório. De seguida, estes valores são divididos por 100000, para ser determinado o tempo médio de inserir uma unidade.

Inserir Localidade (100000)				
	5000	10000	15000	18000
	ns	ns	ns	ns
Média	9,01	9,4	10,45667	13,93667
Mediana	8,55	8,6	9,45	11,7
Moda	7,9	6,3	12,5	11
Desvio-padrão	3,545502	4,3909	4,70962	5,726134

Tabela 33 – Inserção Localidades Teste 3

Inserir Localidade (100000)				
	5000	10000	15000	18000
	ns	ns	ns	ns
Média	17,89	18,04333	19,50667	20,70333
Mediana	17,15	18,6	20,2	20,3
Moda	15,6	18,7	24,9	20,3
Desvio-padrão	3,09798	3,24852	3,8762	4,1084

Tabela 34 – Inserção Localidades Teste 4

À semelhança das procuras, os resultados dos testes de inserção surpreenderam-nos, apesar de, mais uma vez, se esperar por melhores resultados no teste 3. O teste 4 apresentou-se mais rápido. No entanto, ambos apresentaram resultados bastante bons.

Teste 15 – Comparação Inserção Localidades Teste 3 vs Teste 4

3.4.3 Inserir Ligação

Na inserção de ligações, é inserida sempre a mesma ligação, em localidades aleatórias já inseridas na lista de localidades. Desta forma, são inseridas 100000 ligações. De seguida, estes valores são divididos por 100000, para ser determinado o tempo médio de inserir uma unidade.

Inserir Ligação (100000)				
	5000	10000	15000	18000
	ns	ns	ns	ns
Média	6,18	7,013333	7,076667	7,66
Mediana	6,2	7	7,75	7,75
Moda	4,6	6,2	7,9	7,8
Desvio-padrão	2,130792	2,174576	2,700196	2,192872

Tabela 35 – Inserção Ligação Teste 3

Inserir Ligação (100000)				
	5000	10000	15000	18000
	ns	ns	ns	ns
Média	11,94667	13,75	14,49	15,18
Mediana	11	14	14,1	14,85
Moda	10,9	14	15,6	12,4
Desvio-padrão	2,46223	2,014901	2,749626	3,085606

Tabela 36 – Inserção Ligação Teste 4

À semelhança dos testes anteriores, verifica-se que o teste 4 supera o teste 3, no entanto, mais uma vez, verifica-se que os Maps apresentam velocidades bastante melhores que as restantes estruturas, neste tipo de comparação.

Ilustração 16 – Comparação Inserir Ligação Teste 3 vs Teste 4

3.4.4 Ler ficheiro

Nesta funcionalidade, como os tempos são maiores que os restantes, mas continuando com valores pequenos, só foi necessário fazer 10 leituras. Contudo, também são realizados 30 ensaios.

Ler ficheiro (10)				
	5000	10000	15000	18000
	ms	ms	ms	ms
Média	76,12333	95,15	115,64	126,9733
Mediana	74,9	95,15	115,4	126,4
Moda	74,9	95,2	115,4	127,9
Desvio-padrão	2,947902	1,414396	1,421752	1,39307

Tabela 37 – Ler ficheiro Teste 3

Ler ficheiro (10)				
	5000	10000	15000	18000
	ms	ms	ms	ms
Média	76,9	99,83333	126,2433	140,1267
Mediana	76,4	99,8	126,3	140,35
Moda	76,4	99,8	124,8	140,4
Desvio-padrão	1,168849	0,917393	1,87173	1,778014

Tabela 38 – Ler ficheiro Teste 4

Podemos concluir, através das tabelas, que esta funcionalidade é mais rápida no Teste 1. No entanto, visto que, ao ler um ficheiro vão ser adicionados clientes, estes tempos vão ser quase proporcionais aos tempos das inserções de localidades.

Nas inserções o TreeMap superava o HashMap, e, no entanto, é normal que aqui também supere e os tempos sejam quase proporcionais.

Percorrer e Imprimir				
	5000	10000	15000	18000
	s	s	s	s
Média	45,14967	55,261	67,821	74,15667
Mediana	45,005	55,14	67,935	73,705
Moda	44,3	54,75	66,14	73,16
Desvio-padrão	1,219328	1,361291	1,375899	1,297303

Tabela 39 – Percorrer e Imprimir Localidade Teste 3

Percorrer e Imprimir				
	5000	10000	15000	18000
	s	s	s	s
Média	45,14967	55,261	67,821	74,15667
Mediana	45,005	55,14	67,935	73,705
Moda	44,3	54,75	66,14	73,16
Desvio-padrão	1,219328	1,361291	1,375899	1,297303

Tabela 40 – Percorrer e Imprimir Localidades Teste 4

stração 17 – Comparação Ler ficheiro Teste 3 vs Teste 4

3.4.5 Percorrer e Imprimir

Nesta funcionalidade, como os tempos são mais elevados não é necessário fazer mais que uma listagem. Contudo, também são realizados 30 ensaios.

Analisando os gráficos, verifica-se que ambos os testes apresentam tempos bastante semelhantes. Desta vez, ao contrário dos outros testes, verifica-se que as estruturas em Map são piores para situações de percorrer e imprimir.

Ilustração 18 - Comparação Percorrer e Imprimir Teste 3 vs Teste 4

3.5 Leitura e escrita

Teste 1: streams de texto, para leitura (BufferedReader) e para escrita (PrintWriter);

Teste 2: streams de objecto, para leitura (ObjectInputStream) e para escrita (ObjectOutputStream);

Ambos os testes são baseados em estruturas TreeMap.

3.5.1 Leitura de Localidades

Na leitura de localidades em modo texto são lidas dez vezes o mesmo ficheiro. De seguida, os valores obtidos são divididos por dez, para ser determinado o tempo médio de inserir uma unidade. Por outro lado, em modo binário, o ficheiro só é lido uma vez. Isto deve-se ao facto de, um dos testes ser mais lento do que outro.

Ler Localidades em modo texto (10)				
	5000	10000	15000	18000
	ds	ds	ds	ds
Média	8,32866	16,343	28,5496	29,6226
Mediana	7,96	16,07	28,785	29,17
Moda	7,8	15,91	27,45	29,17
Desvio-padrão	0,68622	0,69681	1,411305	1,119418

Tabela 41 – Leitura Localidades Teste 1

Ler Localidades em modo binário				
	5000	10000	15000	18000
	ds	ds	ds	ds
Média	120,27	271,4	379,3233	446,8667
Mediana	117	269,85	372,05	439,9
Moda	117	234	358,8	444,6
Desvio-padrão	11,4242	38,7351	24,76619	31,6611

Tabela 42 – Leitura de Localidades Teste 2

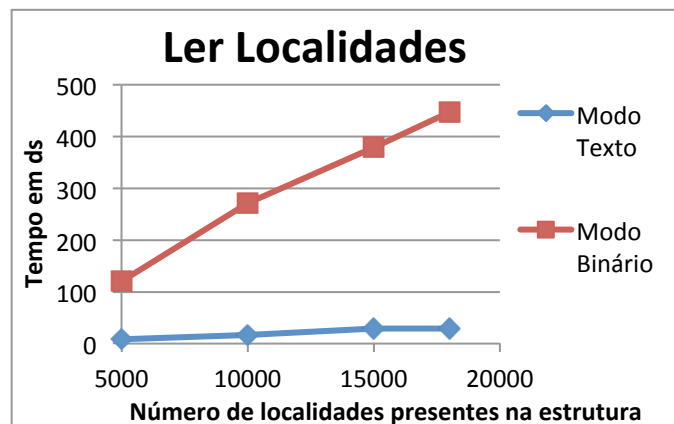


Ilustração 19 – Comparação Leitura localidades Teste 1 vs Teste 2

Analisando o gráfico, verifica-se que a leitura de localidades no Teste 1 apresenta tempos inferiores ao Teste 2. Estes tempos não eram os esperados, uma vez que o Teste 2 deveria apresentar tempos menores que o Teste 1, uma vez que, em modo binário não existe parsing do texto nem inserções.

3.5.2 Escrita de Localidades

Na escrita de localidades em modo texto são escritas dez vezes o mesmo ficheiro. De seguida, os valores obtidos são divididos por dez, para ser determinado o tempo médio de inserir uma unidade. Por outro lado, em modo binário, o ficheiro só é escrito uma vez. Isto deve-se ao facto de, um dos testes ser mais lento do que outro.

Escrever Localidades em modo texto (10)				
	5000	10000	15000	18000
	cs	cs	cs	cs
Média	7,01466	13,306	20,7523	24,7716
Mediana	6,865	13,025	20,75	24,805
Moda	6,71	12,48	20,75	24,96
Desvio-padrão	0,54079	1,13014	1,284657	1,346702

Tabela 43 – Escrita Localidades Teste 1

Escrever Localidades em modo binário				
	5000	10000	15000	18000
	CS	CS	CS	CS
Média	99,2133	185,84	277,093	334,133
Mediana	96,7	181,75	273	332,25
Moda	96,7	177,8	273	318,2
Desvio-padrão	9,863053	11,63119	11,9829	22,0395

Tabela 44 – Escrita Localidades Teste 2

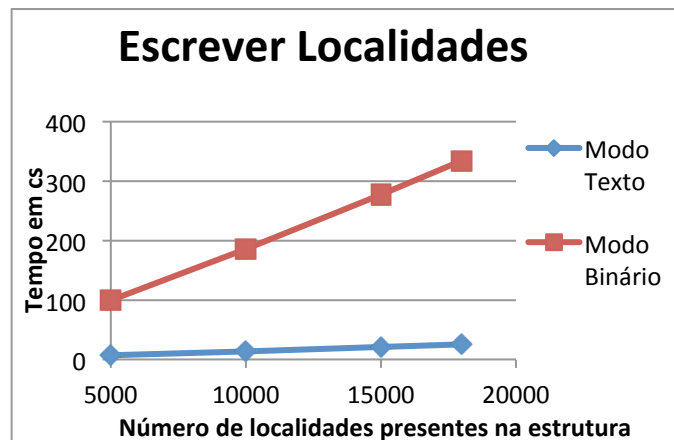


Ilustração 20 – Comparação escrita localidades Teste 1 vs Teste 2

Pela análise do gráfico, conclui-se que o Teste 1 apresenta tempos inferiores. Os resultados obtidos também não foram os esperados, da mesma forma que acontece com a leitura de localidades.

3.5.3 Leitura de Clientes

A leitura de clientes faz-se, tanto para modo binário como para modo texto, da mesma forma que a leitura de localidades.

Ler Clientes em modo texto (10)				
	5000	10000	15000	18000
	CS	CS	CS	CS
Média	1,440333	3,037	4,399	6,125333
Mediana	1,4	2,81	4,29	6,16
Moda	1,4	2,81	4,21	6,24
Desvio-padrão	0,18650	0,41512	0,36996	0,696532

Tabela 45 – Leitura de clientes Teste 1

Ler Clientes em modo binário				
	5000	10000	15000	18000
	CS	CS	CS	CS
Média	24,96	55,4833	75,45	95,8833
Mediana	24,9	49,9	74,9	89,7
Moda	23,4	49,9	74,9	88,9
Desvio-padrão	1,59061	12,1313	3,806923	11,91111

Tabela 46 – Leitura clientes Teste 2

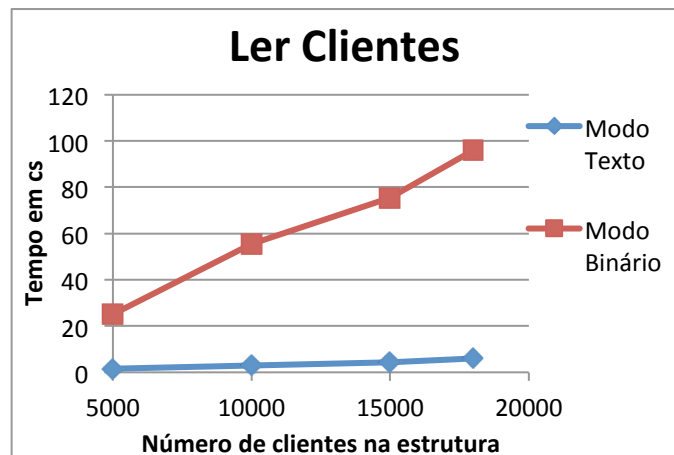


Ilustração 21 – Comparação Leitura Clientes Teste 1 vs Teste 2

Pela análise das tabelas e dos gráficos, pode-se verificar novamente, que os tempos de ambos os testes se assemelham ao tempo das funcionalidades acima e, que uma vez mais, não eram os tempos esperados.

3.5.4 Escrita de Clientes

A escrita de clientes faz-se, tanto para modo binário como para modo texto, da mesma forma que a escrita de localidades.

Escrever Clientes em modo texto (10)				
	5000	10000	15000	18000
	CS	CS	CS	CS
Média	0,83166	1,44533	2,14266	2,829
Mediana	0,78	1,41	2,03	2,8
Moda	0,78	1,4	2,03	2,65
Desvio-padrão	0,07511	0,08135	0,252847	0,374464

Tabela 47 – Escrita Clientes Teste 1

Escrever Clientes em modo binário				
	5000	10000	15000	18000
	ms	ms	ms	ms
Média	14,9733	29,06667	52,88	72,7966
Mediana	14,1	28,1	51,5	73,3
Moda	14	28,1	51,5	73,3
Desvio-padrão	1,72765	1,843971	3,15942	6,29342

Tabela 48 – Escrita Clientes Teste 2

É possível concluir que os tempos da escrita de clientes são superiores no Teste 2. A justificação para tal, assemelha-se às justificações anteriores.

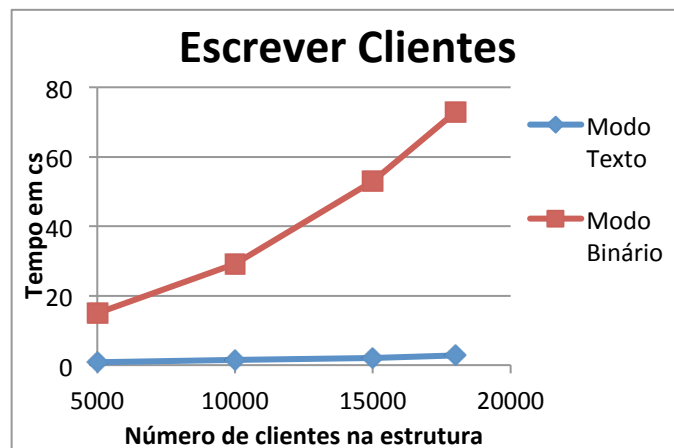


Ilustração 22 – Comparação escrita clients Teste 1 vs Teste 2

4 Estrutura final

Na primeira etapa, foi mencionado que, as estruturas baseadas em `HashMap` eram, na generalidade, as estruturas mais aconselháveis para quase todas as funcionalidades. No entanto, nesta etapa, foi escolhida para estrutura final uma estrutura baseada em `TreeMap`, tanto para clientes como para localidades. Deste modo, a escolha de estruturas baseadas em `Maps` deve-se ao facto de, estas apresentarem tempos mais rápidos, em quase todas as funcionalidades. Por conseguinte, dentro dos `Maps`, foi escolhida a estrutura `TreeMap`, pois estas são as mais aconselháveis para listagens.

Sendo assim, como o programa possui muitas listagens, muitas delas ordenadas por nome, e, como a estrutura `TreeMap` mantém a ordem natural dos dados, esta foi considerada a melhor opção.

5 Interface

Relativamente à interface do programa com o utilizador, primeiramente, é apresentado um menu com duas secções, uma para os clientes e outra para as localidades. Cada uma destas apresenta diversas funcionalidades.

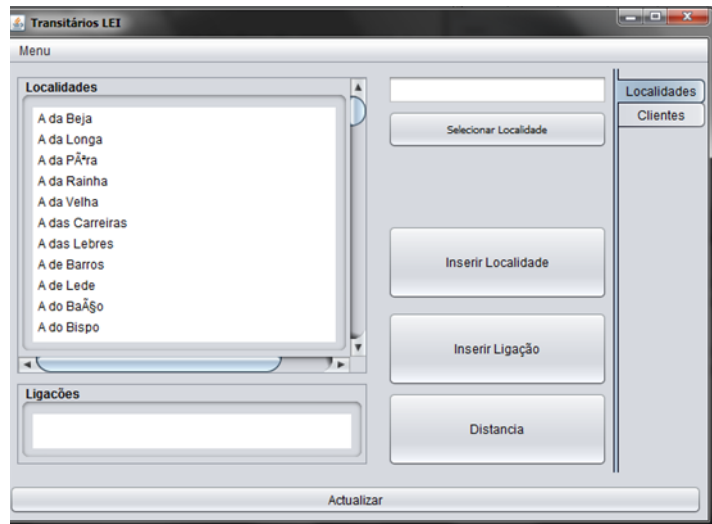


Ilustração 24 – Menu principal - Localidades

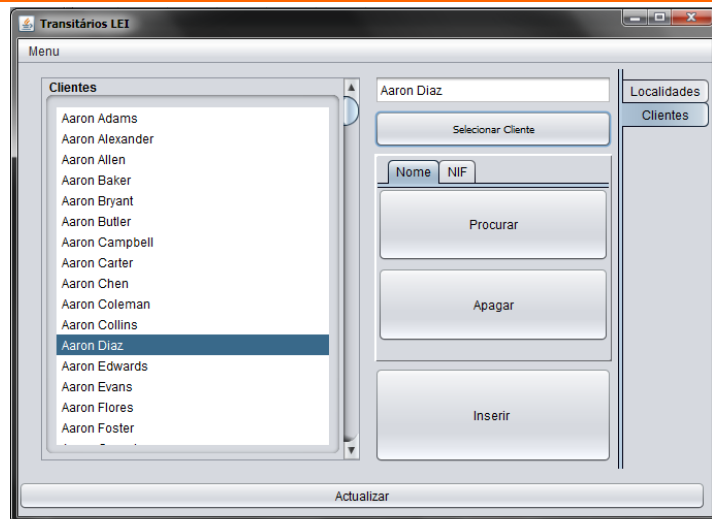



Ilustração 23 – Menu principal – Clientes

Em ambos os menus, existe o botão “Actualizar” que actualiza todo o sistema, nomeadamente os clientes e as localidades.

A caixa de texto visível em ambos os menus é utilizada para escrever o nome da localidade/cliente com que se pretende fazer alguma das funcionalidades, procurar, apagar... Ao invés disso, se seleccionarmos um nome na lista de clientes/localidades este é automaticamente escrito na caixa de texto. Nestes dois casos, ao ser seleccionado um cliente/localidade na



Cliente


Aaron Campbell

900021151

900021151@foursquareUM.com

3310 Harvey Way

Sair



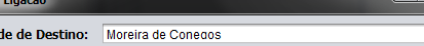
Inserir Localidade

Nome da Localidade:

Codigo Postal:

Inserir

Limpar **Sair**



Inserir Ligação

Localidade de Destino:

Distância : kilómetros

Taxas : euros

Inserir

Limpar **Sair**

Distancia entre duas localidades

Localidade de Destino

- vinha verra
- Vinha da Rainha
- Vinha da Velha
- Vinhais
- Vinhai**
- Vinhas
- Vinhas Novas
- Vinheiros
- Vinho
- Vinhã?
- Vinhã's
- Vinte Seis
- Violeiro

Localidade de Partida

A da Rainha

Localidade de Destino

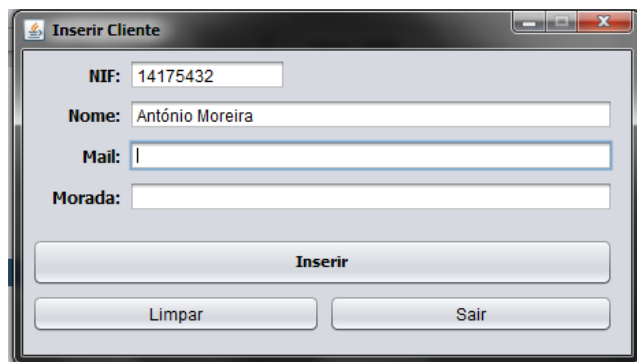
Vinhai

Calcular Distancia

Sair

Ilustração 27 – Menu Calcular distância entre localidades

No menu principal de clientes, existem dois separadores, um para o nome e outro para o NIF. Ambos os separadores possuem as seguintes funcionalidades: procurar, apagar e inserir. Para ser possível apagar ou procurar um cliente é necessário escrever o nome/NIF na caixa de texto, ou seleccionar o cliente na lista de clientes.



The image shows a Windows-style dialog box titled "Inserir Cliente". It contains four text input fields: "NIF:" with the value "14175432", "Nome:" with the value "António Moreira", "Mail:" which is empty and has a blue border, and "Morada:" which is empty. Below the fields are three buttons: "Inserir" (centered), "Limpar" (left), and "Sair" (right).

Ilustração 29 – Menu Inserir cliente

6 Conclusão

Com a recolha de todos os resultados obtidos, pode-se concluir que os Maps são muito mais rápidos em quase todas as operações, no entanto, nos testes de Percorrer e Imprimir verificou-se que as estruturas da classe Collection são mais aconselháveis. Contudo, num programa em que as listagens sejam pouco utilizadas, compensa utilizar estruturas em Map.

Os tempos medidos das inserções/procuras nas Maps estão na ordem dos nano segundos, enquanto que as mesmas operações em outras estruturas estão na ordem dos milisegundos.

Pode-se concluir que, na generalidade dos casos, as estruturas em Map são muito mais rápidas que as restantes, e é por isso normal, que a estrutura futura do nosso programa esteja baseada neste tipo de estruturas.

Depois de avaliar os resultados dos testes, verificou-se que, para estrutura final do nosso programa optaríamos: por uma estrutura baseada em TreeMap para clientes, por nome e por NIF, e uma estrutura baseada em TreeMap para as localidades e respectivas ligações.

No entanto, a estrutura final escolhida foi uma estrutura baseada em TreeMap, tanto para clientes como para localidades. Deste modo, a escolha de estruturas baseadas em Maps já foi referida anteriormente. E, por conseguinte, dentro dos Maps, foi escolhida a estrutura TreeMap, pois estas são as mais aconselháveis para listagens.

Sendo assim, como o programa possui muitas listagens, muitas delas ordenadas por nome, e, como a estrutura TreeMap mantém a ordem natural dos dados, esta foi considerada a melhor opção.

Relativamente aos testes de leitura e escrita entre streams de texto e de objecto, estes apresentaram tempos dos quais não se estavam à espera. Verificou-se que os tempos de streams de texto foram melhores que os tempos de streams de objectos. Estes resultados deviam ser inversos, no entanto, não encontramos razão para tal, provavelmente um erro nosso.

Para finalizar, a interface criada é bastante amigável para o utilizador, e permite uma utilização rápida e é adaptada às diversas funcionalidades.

7 Informação do grupo

7.1 Contributos

André Santos: Responsável pela maior parte do código do programa.

Numa primeira etapa realizou grande parte das classes base, e respectivos métodos base, edição dessas classes para as diferentes estruturas testadas e realização do código dos testes.

De seguida responsável pela criação dos diferentes métodos das classes. Consequentemente realizou o código da leitura e escrita de ficheiros, tanto em modo de texto como em binário.

Responsável pela criação da interface em Swing e respectivo código de ligação com as outras classes.

Percentagem de trabalho total: ~50%

Helena Alves: Numa primeira fase foi responsável pela análise de tempos (criação de tabelas e respectivos gráficos).

De seguida foi Responsável pelos testes da camada de persistência. E respectiva análise destes.

Responsável pelos dois relatórios.

Percentagem de trabalho total: ~25%

Ricardo Branco: Criação de algumas classes base para a primeira entrega, criação dessas mesmas classes com as variáveis de instância e métodos base, como equals, toString, clone e hashCode.

Posteriormente encarregue de mais alguns métodos adicionais como comparadores, adições e remoções.

Na Segunda etapa introduziu uma imagem no carregamento do programa, e introduziu a funcionalidade de através da tecla Enter e do duplo clique abrir as informações da localidade/cliente.

Encarregue também de testes e pequenas correcções no programa.

Percentagem de trabalho total: ~25%

7.2 Fotos



André Santos
A60994



Helena Alves
A61000



Ricardo Branco
A61075

7.3 Informação

André Santos: andreccdr@gmail.com.

Escola EB 2,3 Ribeirão.

Escola Secundária D. Sancho I, Famalicão – Ciências e Tecnologias.

Interesse nas áreas de programação móvel, programação web, gaming, novas tecnologias, informática em geral.

Helena Alves: helenalves4@gmail.com.

Escola EB 2,3 S. Paio Moreira de Cónegos.

Escola Secundária Caldas de Vizela – Ciências e Tecnologias.

Interesse em atletismo e desporto em geral.

Ricardo Branco: ricax92@gmail.com.

Escola EB 2,3/S Pintor José de Brito, Viana do Castelo.

Escola Secundária de Monserrate, Viana do Castelo – Técnico Profissional de Gestão de Equipamentos Informáticos.

Interesse em redes, novas tecnologias e xadrez.