|  |  |  |
| --- | --- | --- |
|  |  |  |
| http://docplayer.com.br/docs-images/21/1172167/images/1-0.png | |  |
| |  |  | | --- | --- | |  |  | |  |  | |  |  | |  |  |   Grupo 7, 2º Semestre 2016  Projeto JAVA - GereVendas  Gestão das Vendas de uma Cadeia de Distribuição | | |

Laboratórios de Informática III

- Alexandre Teixeira (A73547)



- Bruno Sousa (A74330)



- Rafael Silva (A74264)



Índice

[Índice 2](#_Toc453795469)

[*1-* *Estrutura de Dados* 3](#_Toc453795470)

[*1.1-* *Hipermercado* 3](#_Toc453795471)

[1.1.1- Clientes 3](#_Toc453795472)

[1.1.2- Produtos 4](#_Toc453795473)

[1.1.3- Vendas 4](#_Toc453795474)

[*1.2-* *Main* 5](#_Toc453795475)

[*2- Medidas de Perfomance* 5](#_Toc453795476)

[*2.1- T*empos de leitura 5](#_Toc453795477)

[2.1.1- Usando Scanner() e BufferedReader() 6](#_Toc453795478)

[*2.1.2-* *Leitura para ArrayList com parsing* 6](#_Toc453795479)

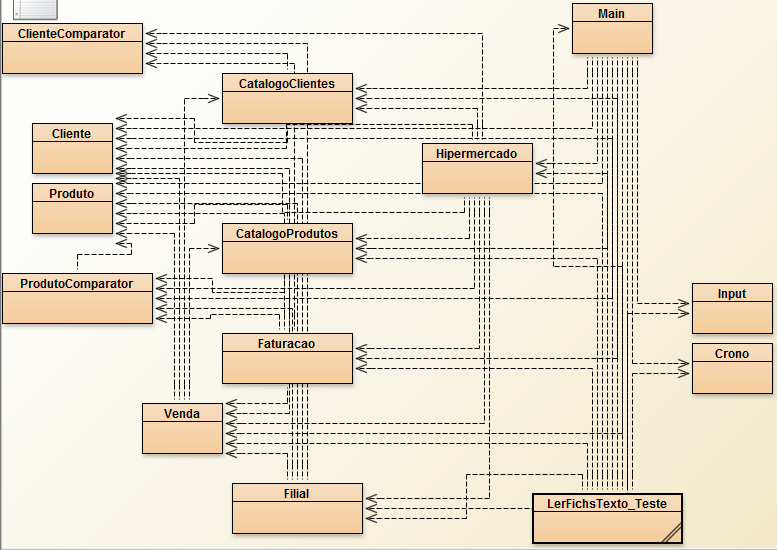
[*2.1.3-* *Leitura para HashSet com parsing* 7](#_Toc453795480)

[*2.1.4-* *Leitura para ArrayList sem parsing* 7](#_Toc453795481)

[*Conclusão* 8](#_Toc453795482)

# *Estrutura de Dados*

## *Hipermercado*



A classe Hipermercado contêm todos os “módulos” para a gestão, correta da aplicação criada. Sendo estes: um Catalogo de Clientes, um Catalogo de Produtos, uma Faturação e três Filiais.

### Clientes

#### Classe Cliente

A classe Cliente contêm a definição de um cliente. Sendo este definido por apenas uma String, para o programa ter uma maior generalidade.

#### Classe ClienteComparator

A classe ClienteComparator têm como finalidade a comparação de dois clientes, de forma crescente segundo o seu nome.

#### Classe CatalogoClientes

A classe CatalogoClientes representa a definição de um catalogo de Clientes, sendo este representado por um Set de clientes (Set<Cliente>).

### Produtos

#### Classe Produto

A classe Produto contêm a definição de um produto. Sendo este definido por apenas uma String, para o programa ter uma maior generalidade.

#### Classe Produto Comparator

A classe ProdutoComparator têm como finalidade a comparação de dois produtos, de forma crescente segundo a sua identificação.

#### Classe CatalogoProdutos

A classe CatalogoProdutos representa a definição de um catalogo de Produtos, sendo este representado por um Set de produtos (Set<Produto>).

### Vendas

#### Classe Venda

A classe Venda contêm a definição de uma venda. Sendo esta definida por Produto, preço, quantidade, tipo (Normal/Promoção), Cliente, mes e filial.

#### Classe Faturacao

A classe Faturacao contêm um Map<Integer,Map<Produto,List<Venda>>>, que relaciona os produtos e as suas vendas, filial a filial e globalmente. O Integer do primeiro Map indica a filial em questão e o segundo Map relaciona cada produdo com a sua lista de vendas.

#### Classe Filial

A classe Filial contêm um int que indica a filial em questão e ua lista de tamanho 12(meses) em que cada elemento contem um Map relacionando os clientes as suas listas de vendas.

## *Main*

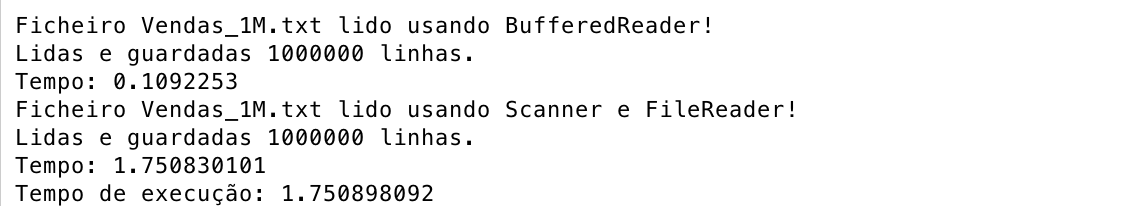


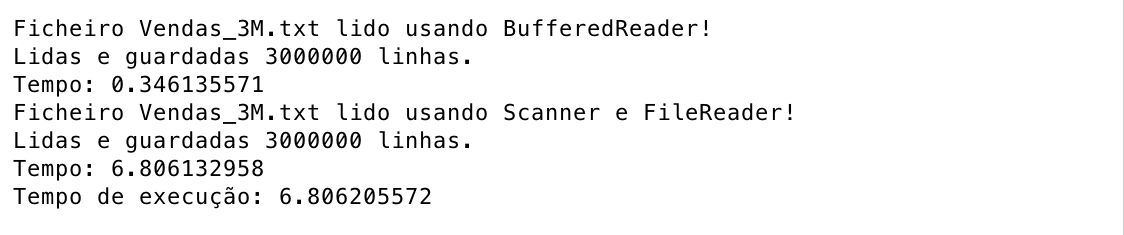
A classe Main implementa o método main que por default carrega os ficheiros “Clientes.txt”, “Produtos.txt” e “Vendas\_1M.txt” e chama o método interpretador, dando-lhe estes três ficheiros. O método interpretador apresenta um menu, com as queries pedidas e outras opções, chamando o método auxiliar correspondente à opção escolhida.

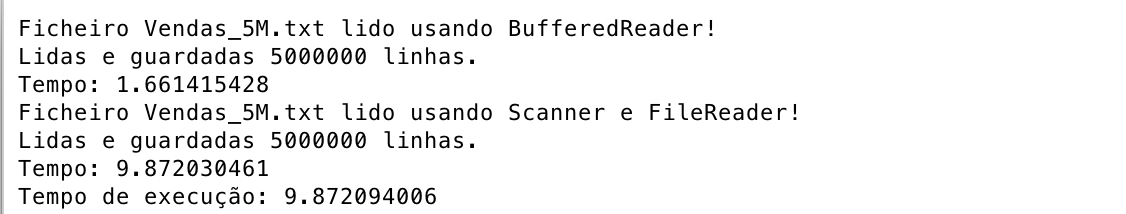
# *2- Medidas de Perfomance*

## *2.1- T*empos de leitura

### Usando Scanner() e BufferedReader()

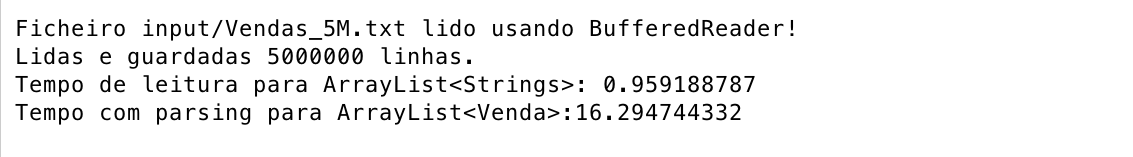
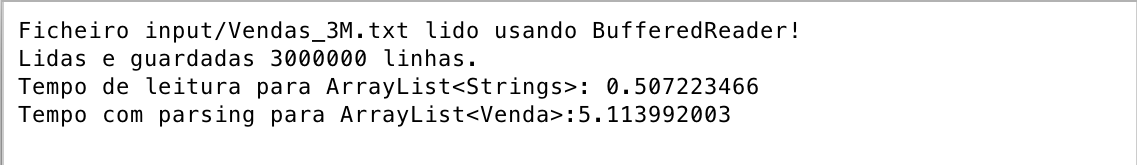
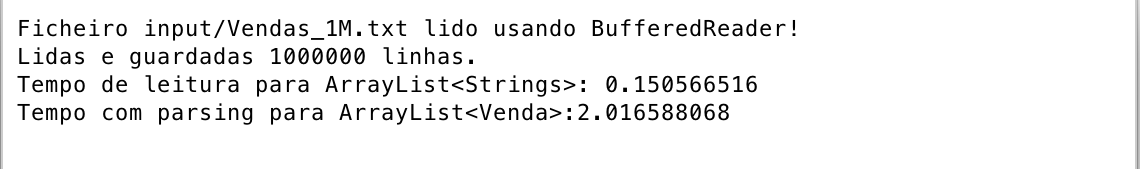
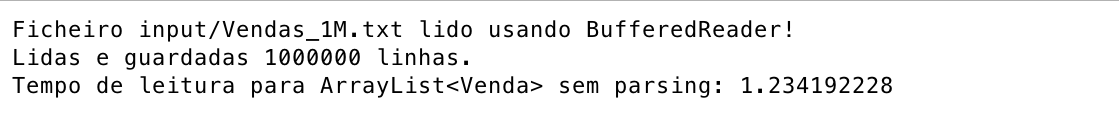


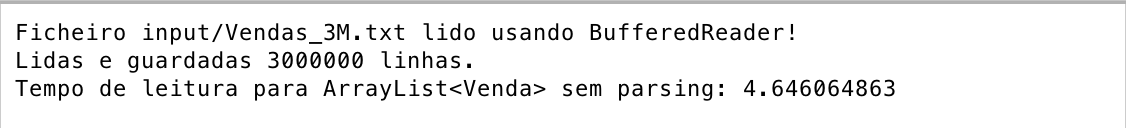
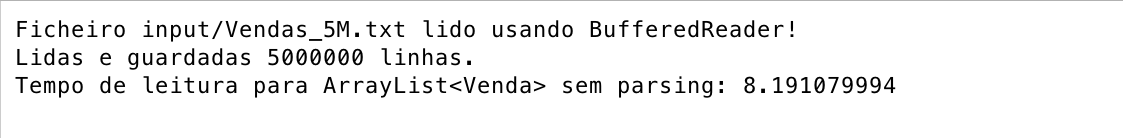




Verificamos que independentemente do tamanho do ficheiro, usando o BufferedReader a leitura é significativamente mais rápida.

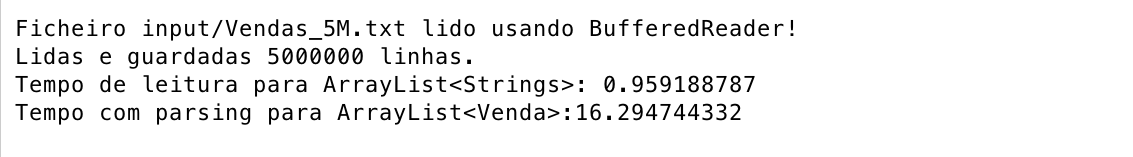
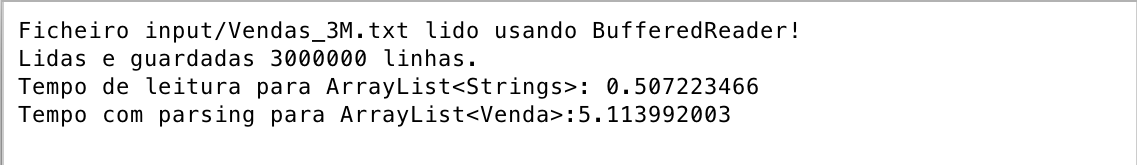
### *Leitura com/sem parsing*





Apesar de não ser tão significativo em ficheiros pequenos, podemos observar que a leitura sem parsing em ficheiros grandes é bastante mais rápida.

### C:\Users\Tatiana Ferreira\Desktop\Tempos\tarefa 6\HashSet\Screen Shot 2016-06-09 at 17.01.12.png*C:\Users\Tatiana Ferreira\Desktop\Tempos\tarefa 6\HashSet\Screen Shot 2016-06-09 at 17.00.05.png*C:\Users\Tatiana Ferreira\Desktop\Tempos\tarefa 6\HashSet\Screen Shot 2016-06-09 at 16.59.30.pngC:\Users\Tatiana Ferreira\Desktop\Tempos\tarefa 6\ArrayList\Screen Shot 2016-06-09 at 16.18.55.png*Leitura para HashSet/ArrayList com Parsing*



Para ficheiros pequenos a diferença é insignificante, no entanto, quando os ficheiros têm um tamanho significativo, dada a sua organização para a leitura o ArrayList é muito mais rápido.

# *Conclusão*

Neste projeto, devido à carga de trabalhos e frequências, por falta de tempo, não nos foi possível implementar todas a queries, tendo ficado por realizar a 7, 8 e 9. No entanto, conseguimos perceber a importância na escolha de certas estruturas de dados, devido as suas performances. Também notamos uma maior facilidade da implementação destas estruturas em relação ao projeto em C.

Neste projeto viu-se a complexidade que é fazer uma gestão de uma cadeia de hipermercados, pois esta trabalha com uma grande extensão de dados, notou-se que foi um pouco mais fácil a implementação em JAVA do que em C, visto que JAVA é uma linguagem mais simples de entender, apesar do projeto ser difícil de implementar em qualquer linguagem.