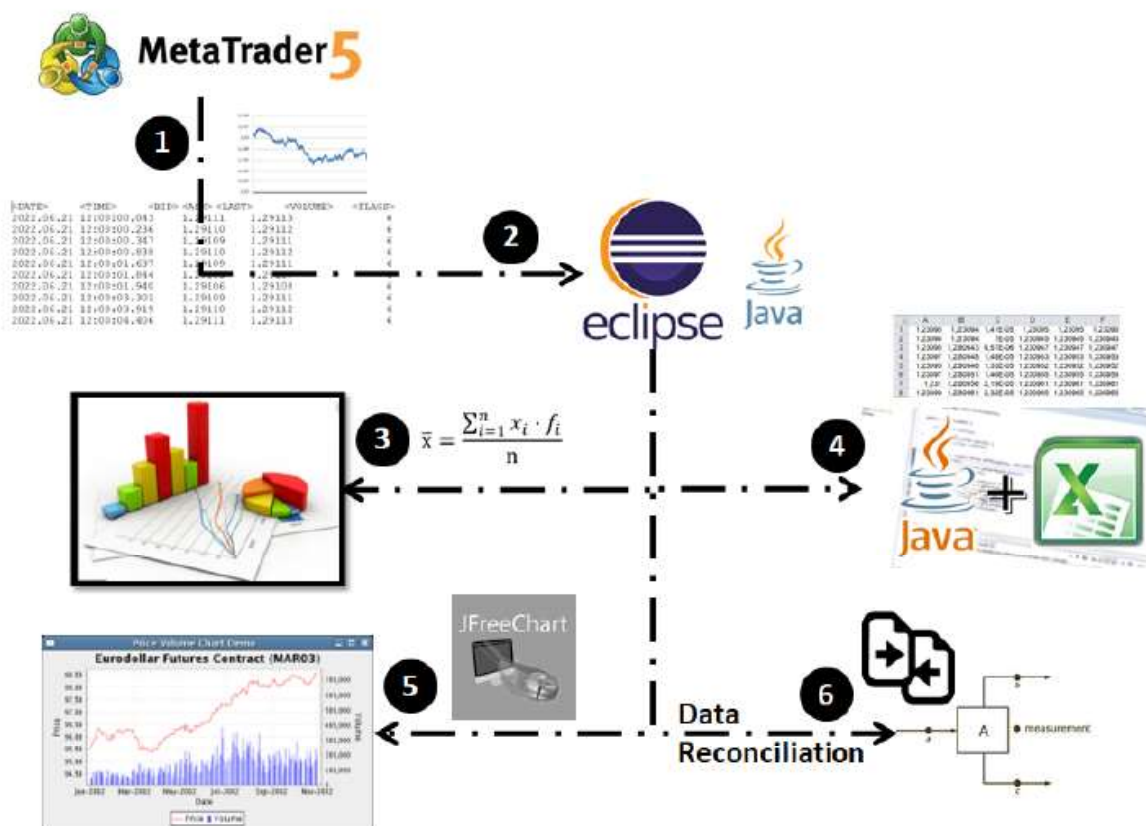


## ➤ Contexto da Aplicação:

Extrair dados de cotações de moedas ou ativos, e realizar o processamento utilizando recursos estatísticos, para obter indicadores de um ativo continuamente durante períodos definidos. Com isso, viabilizar a realização de análises acerca do comportamento do mercado, de forma a auxiliar na leitura de tendências e posteriormente na tomada de decisões, acerca da compra ou venda de um ativo.

Figura 1: Estratégia e recursos utilizados para desenvolver a aplicação proposta.



Fonte: GAT 108 – Automação Avançada – Avaliação 1 (AV1).

## ➤ Resultados e Discussões:

### Parte 1:

Foram desenvolvidos os passos 1 a 5 da aplicação proposta, utilizando o framework Eclipse em conjunto com a linguagem de programação JAVA e conceitos de Orientação a Objetos.

Inicialmente, criou-se a classe Ativo para representar um ativo, em seguida criou-se a classe LeituraArquivoCSV que estabelece uma relação de composição com a Ativo, e utilizou-se a biblioteca OpenCSV que oferece suporte a todas as operações básicas do tipo CSV para JAVA. Posteriormente, criou-se a classe Processamento que estabelece uma relação de agregação com a classe de leitura de arquivos CSV, e uma relação de associação com a classe Ativo. Essa classe de Processamento processa os dados e calcula os elementos estatísticos requisitados.

A partir daí, criou-se as classes RegistroExcel e Grafico que herdam da classe Processamento, nessas subclasses utilizou-se as API's JExcel e JFreeChart para viabilizar a geração de arquivos excel e gráficos respectivamente.

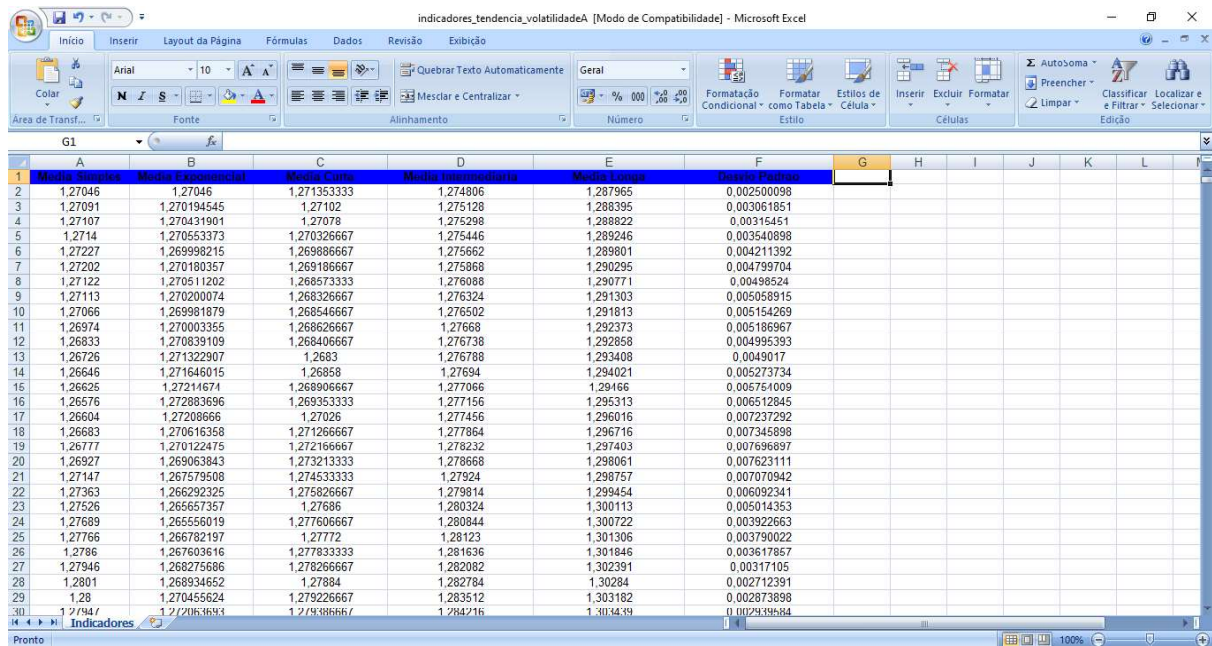
As figuras a seguir mostram os resultados obtidos na realização de testes com o programa desenvolvido.

Figura 2: código da classe Principal2 chamando métodos para construção do arquivo excel e gráficos.

```
1 package aplicacao;
2
3 /*
4  * Classe Principal2, contém o método Main
5  * será utilizada para executar o programa e
6  * gerar arquivos Excel e Gráficos a partir
7  * dos dados passados como parâmetros.
8  *
9  * @author Rugelli Oliveira
10  * @version 1.0
11  */
12
13 public class Principal2 {
14
15     public static void main(String[] args) {
16
17         //Gerar gráficos para cada Ativo
18         Grafico graficoA = new Grafico("USDCAD_AtivoA.csv", "Ativo A - (US Dollar vs Canadian Dollar)");
19         graficoA.exibir();
20
21         //Gerar arquivos excel para cada Ativo
22         RegistroExcel excelA = new RegistroExcel("USDCAD_AtivoA.csv", "C:\\Users\\Rugelli\\Desktop\\"
23             + "Indicadores_tendencia_volatilidadeA.xls", "Indicadores");
24         excelA.gerarRegistro();
25     }
26 }
```

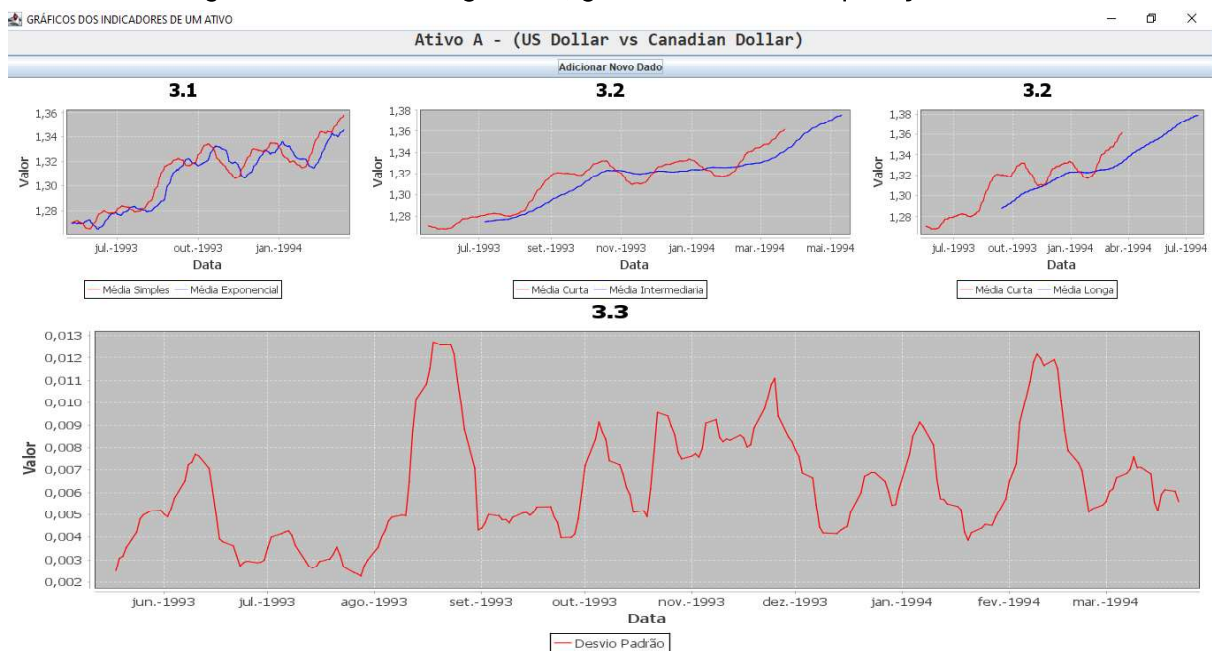
Fonte: Elaborado pelo autor.

Figura 3: Arquivo Excel gerado através da aplicação JAVA.



Fonte: Elaborado pelo autor.

Figura 4: Janela com gráficos, gerada através da aplicação JAVA.



Fonte: Elaborado pelo autor.

No que diz respeito a janela com os gráficos, foi inserido um botão “Adicionar Novo Dado” e configurado um evento de clique, para que a cada clique no botão um novo dado calculado será plotado no gráfico.

Analisando os resultados obtidos, percebe-se que estão de acordo com o esperado, a planilha contém uma coluna para cada tipo de item calculado e os dados são plotados nos gráficos de modo dinâmico.

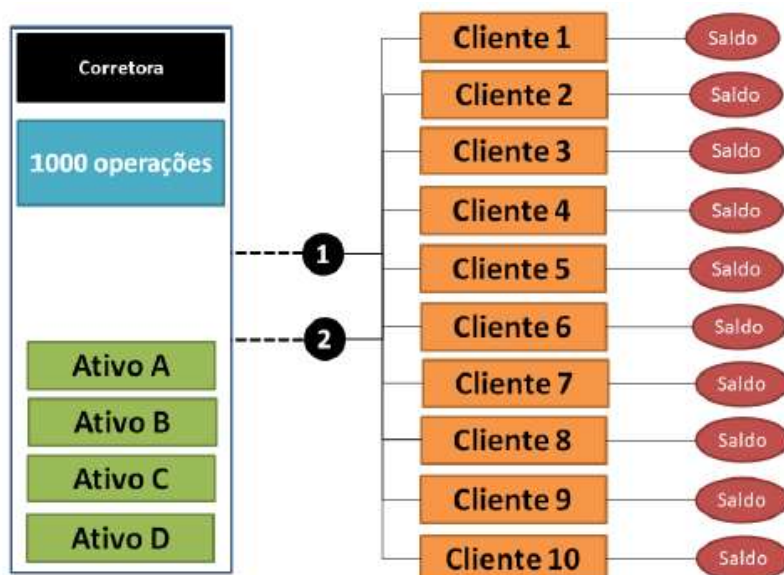
Observa-se também que, a partir da média móvel exponencial é possível acompanhar a mudança do preço de um ativo com mais rapidez, quando comparado com a média móvel simples para um mesmo número de períodos.

Além disso, a média móvel intermediária e a média móvel longa apresentam um atraso em relação à média móvel curta, devido à diferença entre o número de períodos de cada uma.

Por último, a curva do desvio padrão (3.3), demonstra o quão amplamente os preços estão dispersos do preço médio, indicando a volatilidade do mercado. Por exemplo, se compararmos o período entre julho-1993 e ago-1993 e o período entre ago-1993 e set-1993, nota-se que o primeiro indica uma baixa volatilidade em relação ao segundo.

## Parte 2:

Figura 5: Esquema da relação entre a corretora de ativos e os clientes.



Fonte: GAT 108 – Automação Avançada – Avaliação 1 (AV1).

Dando continuidade ao que foi implementado na parte 1 deste trabalho, criou-se a classe Corretora implementando a interface Runnable para fazer dela uma Thread e possibilitar o uso dos conceitos de programação concorrente. Utilizou-se também o conceito de semáforo, para simular e controlar o acesso de vários clientes aos dois caixas da Corretora.

Para isso, criou-se a classe Cliente utilizando o conceito de herança para fazer dela uma Thread através do comando “extends Thread”.

Entre as classes Corretora e Cliente há relações de dependência e associação, pois um objeto utiliza os recursos de outro em momentos específicos.



A classe Corretora disponibiliza aos clientes os timestamp's (datas) e os respectivos valores/preços dos ativos. Já na classe Cliente, são implementados os métodos para consultar as informações recebidas da corretora, e também os métodos para analisar os indicadores de risco e volatilidade, por fim implementou-se os critérios de compra e venda, utilizando mecanismos de drawdown para controlar o risco, e a cada operação finalizada pelo cliente o saldo do mesmo sofre um aumento ou redução de acordo com o tipo da movimentação realizada.

Para registrar as operações dos clientes e da corretora, criou-se as classes ContaCorrente e CaixaGeral respectivamente, as quais são responsáveis por registrar informações referentes aos ativos e às operações realizadas em arquivos texto (.txt), que serão úteis na etapa de reconciliação de dados.

As figuras a seguir mostram os resultados obtidos na realização de testes com o programa desenvolvido.

Figura 6: código da classe Principal chamando métodos para simular os clientes acessando os caixas da corretora.

```
1 package aplicacao;
2
3 import java.util.ArrayList;
4
5
6 /*
7  * Classe Principal, contém o método Main
8  * será utilizada para executar o programa e simular
9  * os acessos de Clientes aos caixa da Corretora.
10  *
11  * @author Rugelli Oliveira
12  * @version 1.0
13  */
14
15 public class Principal {
16
17     public static void main(String[] args) throws Exception {
18
19
20         int numeroDePermissoes = 1;
21         Semaphore semaphore = new Semaphore(numeroDePermissoes);
22         Semaphore semaphore1 = new Semaphore(numeroDePermissoes);
23
24         Corretora c = new Corretora("Avançada", new CaixaGeral("Avançada"), "USDCAD_AtivoA.csv", "EURUSD_AtivoB.csv",
25             "USDJPY_AtivoC.csv", "USDCHF_AtivoD.csv", semaphore, semaphore1);
26
27         ArrayList<Cliente> clientes = new ArrayList<>();
28
29         clientes.add(new Cliente("José", new ContaCorrente(900.0), c));
30         clientes.add(new Cliente("João", new ContaCorrente(800.0), c));
31         clientes.add(new Cliente("Maria", new ContaCorrente(700.0), c));
32         clientes.add(new Cliente("Ana", new ContaCorrente(600.0), c));
33         clientes.add(new Cliente("Joaquim", new ContaCorrente(500.0), c));
34         clientes.add(new Cliente("Mariana", new ContaCorrente(2000.0), c));
35         clientes.add(new Cliente("Sebastião", new ContaCorrente(3000.0), c));
36         clientes.add(new Cliente("Renata", new ContaCorrente(4000.0), c));
37         clientes.add(new Cliente("Antônio", new ContaCorrente(5000.0), c));
38         clientes.add(new Cliente("Beatriz", new ContaCorrente(6000.0), c));
39
40         for(int i = 0; i < clientes.size(); i++) {
41
42             clientes.get(i).start();
43
44         }
45     }
46 }
```

Fonte: Elaborado pelo autor.

Figura 7: Resultados obtidos através da execução do código apresentado na figura anterior.

```
***** SEJAM BEM-VINDOS À CORRETORA AVANÇADA! *****

Cliente: Ana acessou o CAIXA #1
Saldo atual: 600.0
Comprou o ativo C
Saldo atual: 475.76
Cliente: Ana saiu do CAIXA #1

Cliente: Mariana acessou o CAIXA #2
Saldo atual: 2000.0
Comprou o ativo C
Saldo atual: 1875.76
Cliente: Mariana saiu do CAIXA #2

Cliente: Joaquim acessou o CAIXA #2
Saldo atual: 500.0
Comprou o ativo C
Saldo atual: 375.76
Cliente: Joaquim saiu do CAIXA #2

Cliente: Renata acessou o CAIXA #1
Saldo atual: 4000.0
Comprou o ativo C
Saldo atual: 3875.76
Cliente: Renata saiu do CAIXA #1

Cliente: José acessou o CAIXA #1
Saldo atual: 900.0
Comprou o ativo C
Saldo atual: 775.76
Cliente: José saiu do CAIXA #1

Cliente: José acessou o CAIXA #2
Saldo insuficiente para comprar o ativo D
Cliente: José saiu do CAIXA #2

Cliente: Renata acessou o CAIXA #2
Saldo atual: 3876.8139000000006
Comprou o ativo A
Saldo atual: 3875.4919000000004
Cliente: Renata saiu do CAIXA #2

Cliente: Maria acessou o CAIXA #1
Saldo atual: 575.4918999999999
Vendeu o ativo A
Saldo atual: 576.8258999999998
Cliente: Maria saiu do CAIXA #1

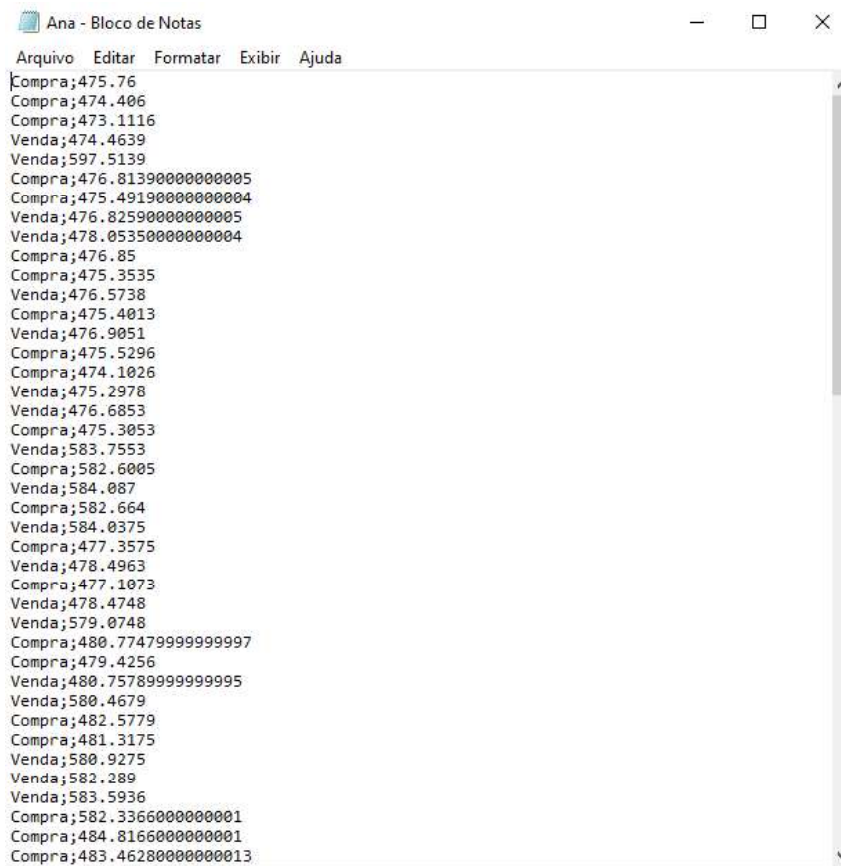
Cliente: Maria acessou o CAIXA #1
Saldo atual: 540.5402000000001
Comprou o ativo B
Saldo atual: 539.3788000000002
Cliente: Maria saiu do CAIXA #1

No momento, a Corretora Avançada não pode realizar mais operações. Obrigado pela compreensão!
```

Fonte: Elaborado pelo autor.

Analisando a figura 7, verifica-se que os caixas 1 e 2 são acessados por um cliente a cada vez, isto é, enquanto um cliente acessa um caixa, outro cliente pode acessar o outro disponível, os demais clientes precisam aguardar até que um dos caixas esteja disponível. Além disso, também é possível observar que quando a corretora atinge o limite de operações realizadas a execução do programa é encerrada.

Figura 8: Dados registrados na conta corrente (arquivo de texto) de um dos clientes.



Ana - Bloco de Notas

Arquivo Editar Formatar Exibir Ajuda

Compra;475.76  
Compra;474.406  
Compra;473.1116  
Venda;474.4639  
Venda;597.5139  
Compra;476.81390000000005  
Compra;475.49190000000004  
Venda;476.82590000000005  
Venda;478.05350000000004  
Compra;476.85  
Compra;475.3535  
Venda;476.5738  
Compra;475.4013  
Venda;476.9051  
Compra;475.5296  
Compra;474.1026  
Venda;475.2978  
Venda;476.6853  
Compra;475.3053  
Venda;583.7553  
Compra;582.6005  
Venda;584.087  
Compra;582.664  
Venda;584.0375  
Compra;477.3575  
Venda;478.4963  
Compra;477.1073  
Venda;478.4748  
Venda;579.0748  
Compra;480.77479999999997  
Compra;479.4256  
Venda;480.75789999999995  
Venda;580.4679  
Compra;482.5779  
Compra;481.3175  
Venda;580.9275  
Venda;582.289  
Venda;583.5936  
Compra;582.33660000000001  
Compra;484.81660000000001  
Compra;483.46280000000013

Fonte: Elaborado pelo autor.





As figuras a seguir apresentam resultados obtidos a partir da realização de testes.

Figura 10: código da classe Principal3 chamando método para apresentar informações referentes ao processo de reconciliação de dados.

```
1 package aplicacao;
2
3 /*
4  * Classe Principal3, contém o método Main
5  * será utilizada para executar o programa
6  * e apresentar informações acerca da reconciliação
7  * de dados.
8  *
9  * @author Rugelli Oliveira
10 * @version 1.0
11 */
12
13 public class Principal3 {
14
15     public static void main(String[] args) {
16
17         Reconciliacao reconciliacao = new Reconciliacao();
18         reconciliacao.exibirInformacoes();
19
20     }
21
22 }
23 }
```

Fonte: Elaborado pelo autor.

Figura 11: Resultados obtidos através da execução do código apresentado na figura anterior.

```
Quantidade de vendas registradas no caixa geral da corretora: 487
Quantidade total de operações de venda realizadas por todos os clientes : 487
Quantidade de compras registradas no caixa geral da corretora: 513
Quantidade total de operações de compra realizadas por todos os clientes : 513

As operações no Caixa Geral da Corretora estão condizentes com as contas-correntes dos 10 Clientes!
```

Fonte: Elaborado pelo autor.

Com base nos resultados mostrados na figura 11, confirma-se que a conferência das informações do caixa geral com as contas correntes dos clientes foi bem sucedida.

## ➤ Conclusão:

Em vista de tudo o que foi apresentado, pode-se constatar que apesar das dificuldades enfrentadas durante a implementação dos códigos para a aplicação JAVA, obteve-se resultados satisfatórios. Optou-se por dividir o trabalho proposto em partes para facilitar a resolução. Na primeira parte

tratou-se a geração de arquivos Excel e gráficos, na segunda parte simulou-se a relação corretora versus clientes já na terceira e última parte implementou a técnica de Reconciliação de Dados com o intuito de verificar se cada operação no Caixa Geral da Corretora condiz com as contas-correntes dos 10 Clientes.

Por fim, vale ressaltar a importância deste trabalho, pois além de ter propiciado colocar em prática conceitos de orientação a objetos e programação concorrente, ele abordou um tema muito interessante que nos permitiu adquirir valiosos conhecimentos acerca do mercado financeiro.

### ➤ **Referências Bibliográficas:**

Silva, Duarte Aurélio Veloso. **Desenvolvimento de um algoritmo para negociação automática no Mercado Cambial (FOREX - Foreign Exchange Market)**. Dissertação (Mestrado) - Instituto Superior de Engenharia do Porto. Departamento de Engenharia Electrotécnica Mestrado em Engenharia Electrotécnica e de Computadores. Porto, 2016.

**Média Móvel: aprenda o que é e como utilizar esse indicador.** Disponível em: <https://blog.toroinvestimentos.com.br/trading/media-movel>. Acesso em: 04/07/2022.

**Médias Móveis (Simples e Exponencial), Cruzamento e Bandas de Bollinger.** Disponível em: <https://proeducacional.com/ead/curso-cga-modulo-i/capitulos/capitulo-4/aulas/medias-moveis-simples-e-exponencial-cruzamento-bandas-de-bollinger/#:~:text=A%20m%C3%A9dia%20m%C3%B3vel%20de%20per%C3%ADodo,20%20dias%2C%20por%20exemplo>. Acesso em: 04/07/2022.

**Desvio padrão - Volatilidade.** Disponível em: <https://www.fidelity.com/learning-center/trading-investing/technical-analysis/technical-indicator-guide/standard-deviation>. Acesso em: 04/07/2022.

**O que é volatilidade?** Disponível em: <https://www.avatradeportuguese.com/education/trading-for-beginners/what-is-volatility>. Acesso em: 04/07/2022.

**O que é risco e retorno em finanças e investimentos?** Disponível em: <https://cienciaenegocios.com/o-que-e-risco-e-retorno-em-financas-e-investimentos/>. Acesso em: 04/07/2022.

**O que é operar comprado.** Disponível em: <https://warren.com.br/magazine/o-que-e-operar-comprado/>. Acesso em: 04/07/2022.

**Drawdown: O que é e por que é importante analisar?** Disponível em: <https://gorila.com.br/blog/drawdown/>. Acesso em: 04/07/2022.

**Ler um arquivo CSV em Java usando OPENCSV.** Disponível em: <https://acervolima.com/ler-um-arquivo-csv-em-java-usando-opencsv/>. Acesso em: 04/07/2022.

**Utilizando JExcelAPI para criar arquivos para o Excel.** Disponível em: <https://www.feltex.com.br/felix/manipulando-planilhas-com-jexcel/>. Acesso em: 04/07/2022.

**JFreeChart 1.0.19 API.** Disponível em: <http://javadocx.com/org.jfree/jfreechart/1.0.19/overview-summary.html>. Acesso em: 11/07/2022.