

Atividade Prática

1) Implementação do método `getVolumeMedio()` da Classe Ação

Definição: Função que calcula o volume médio de uma ação

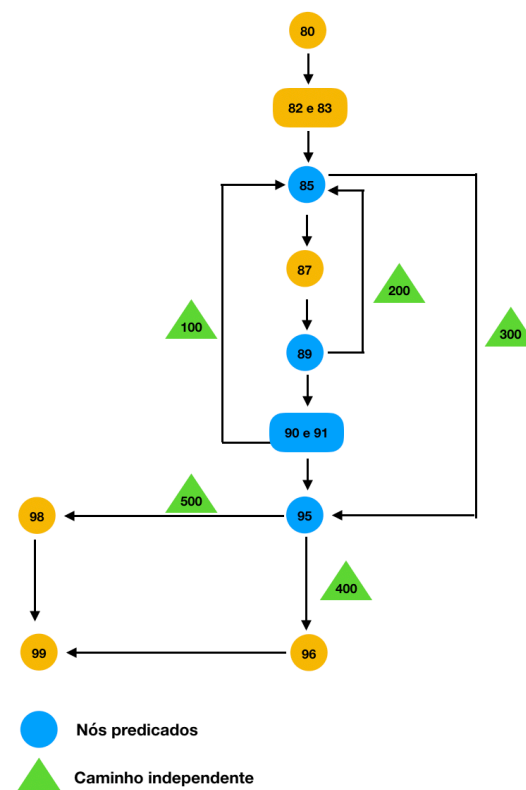
Pre: Não recebe nenhum parâmetro como entrada

Pos: Retorna o volume médio de uma ação com base na somaVolumes e numeroDeFechamentos.






Se o número de fechamentos for igual a zero, o retorno do volume médio será zero.

```
80 public double getVolumeMedio() {  
81  
82     long numeroDeFechamentos = 0;  
83     long somaVolumes = 0;  
84  
85     for (FechamentoAcao fechamentoAcao : fechamentos) {  
86  
87         long volume = fechamentoAcao.getVolume();  
88  
89         if (volume != 0) {  
90             numeroDeFechamentos++;  
91             somaVolumes += volume;  
92         }  
93     }  
94  
95     if (numeroDeFechamentos > 0)  
96         return somaVolumes / (double) numeroDeFechamentos;  
97  
98     return 0d;  
99 }
```

2) Grafo do método `getVolumeMedio()` com os caminhos básicos



- Legenda:

-  Enquanto a variável **fechamentos** não atingir o seu limite, o loop continuará sendo executado
-  Quando a variável **volume = 0**, o código dentro do if não será executado
-  **fechamentos = 0** ou fechamentos chegou ao seu limite depois de todas as execuções do loop
-  **numeroDeFechamentos > 0** o return dentro do if será executado
-  **numeroDeFechamentos < 0**, o return dentro do if não será executado

- Especificação dos caminhos básicos:

	Caminhos	Predicados	Dados
	{80, 82 e 83, 85, 87, 89, 90 e 91, 85 , 95, 96, 99}	p/ todo Fechamentos > 0 p/ todo Volume > 0	Fechamentos = 1 Volume = 100
	{80, 82 e 83, 85, 87, 89, 85 , 95, 98, 99}	p/ todo Fechamentos > 0 p/ todo Volume = 0	Fechamentos = 1 Volume = 0
	{80, 82 e 83, 85, 95 , 98, 99}	p/ todo Fechamentos = 0 p/ todo volume = 0	Fechamentos = 0 Volume = 0
	{80, 82 e 83, 85, 87, 89, 90 e 91, 85, 95, 96, 99 }	p/ todo Fechamentos > 0 p/ todo Volume > 0	Fechamentos = 1 Volume = 100
	{80, 82 e 83, 85, 87, 89, 85, 95, 98, 99 }	p/ todo Fechamentos > 0 p/ todo Volume = 0	Fechamentos = 1 Volume = 0

3) Testes Unitários no método getVolumeMedio()

```

@Before
public void runBefore() {
    acao0GXP3 = new Acao("0GXP3");
    acaoMGLU3 = new Acao("MGLU3");

    LocalDate data1 = LocalDate.parse("2010-01-20");
    BigDecimal valorFechamento1 = new BigDecimal("30.30");
    long volume1 = 34252600;
    acao0GXP3.addFechamento(new FechamentoAcao(acao0GXP3, data1, valorFechamento1, volume1));

    LocalDate data2 = LocalDate.parse("2010-01-21");
    BigDecimal valorFechamento2 = new BigDecimal("20.10");
    long volume2 = 562343;
    acao0GXP3.addFechamento(new FechamentoAcao(acao0GXP3, data2, valorFechamento2, volume2));

    LocalDate data3 = LocalDate.parse("2010-01-22");
    BigDecimal valorFechamento3 = new BigDecimal("25.50");
    long volume3 = 34257500;
    acao0GXP3.addFechamento(new FechamentoAcao(acao0GXP3, data3, valorFechamento3, volume3));

    LocalDate data4 = LocalDate.parse("2010-01-23");
    BigDecimal valorFechamento4 = new BigDecimal("25.50");
    long volume4 = 0;
    acao0GXP3.addFechamento(new FechamentoAcao(acao0GXP3, data4, valorFechamento4, volume4));
}

```

```

85 @Test
86 public void testAcaoVolumeMedio() {
87
88     Double volumeMedio = acaoOGXP3.getVolumeMedio();
89     assertEquals("0 volume médio deve ser 23024148.0", Math.round(23024148.0), Math.round(volumeMedio));
90
91     Double volumeMedioMGLU3= acaoMGLU3.getVolumeMedio();
92     assertEquals("0 volume médio deve ser 0", Math.round(0.0), Math.round(volumeMedioMGLU3));
93 }
94 }

```

O grafo com os caminhos básicos foi criado a partir do método `getVolumeMédio()` e posteriormente todos esses caminhos tiveram que ser analisados para montar os testes unitários.

4) Cobertura de teste do método `getVolumeMedio()`

```

80 public double getVolumeMedio() {
81
82     long numeroDeFechamentos = 0;
83     long somaVolumes = 0;
84
85     for (FechamentoAcao fechamentoAcao : fechamentos) {
86
87         long volume = fechamentoAcao.getVolume();
88
89         if (volume != 0) {
90             numeroDeFechamentos++;
91             somaVolumes += volume;
92         }
93     }
94
95     if (numeroDeFechamentos > 0)
96         return somaVolumes / (double) numeroDeFechamentos;
97
98     return 0d;
99 }

```

A partir da execução de cobertura de testes do Eclipse, podemos ver que todas as áreas da função `getVolumeMedio` foram cobertas.

5) Relatório da execução dos Testes Unitários

Package Explorer JUnit

Finished after 0.136 seconds

Runs: 6/6 Errors: 0 Failures: 0

br.com.empresamodel.AcaoTest [Runner: JUnit 5] (0.069 s)

- testAcaoFechamentoMaximo (0.065 s)
- testAcaoFechamentoMinimo (0.001 s)
- testAcaoFechamentoMaiorRetorno (0.001 s)
- testAcaoVolumeMedio (0.000 s)
- testAcaoFechamentoMenorRetorno (0.000 s)
- testAcaoGetId (0.001 s)

6) Relatório da execução do SonarLint para fazer uma análise estática

Problems

Javadoc

Declaration

Console

SonarLint Report

48 items

Resource	Date	Description
Acao.java		Override "equals(Object obj)" to comply with the contract of the "compareTo(T o)" method.
Acao.java		Replace the type specification in this constructor call with the diamond operator ("<>").
Acao.java		Replace the type specification in this constructor call with the diamond operator ("<>").
Acao.java		Replace the type specification in this constructor call with the diamond operator ("<>").
Acao.java		Replace the type specification in this constructor call with the diamond operator ("<>").
Acao.java		Replace the type specification in this constructor call with the diamond operator ("<>").
Acao.java		Make this anonymous inner class a lambda
Acao.java		Make this anonymous inner class a lambda
Acao.java		Replace this assert with a proper check.

12 files of project quantum-finance-avaliacao (at 02/07/2019 12:59)

7) Relatório da aplicação de Testes de Mutação nos testes do método getVolumeMedio()

Problems	Javadoc	Declaration	Console	PIT Mutations	PIT Summary
----------	---------	-------------	---------	---------------	-------------

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage	Mutation Coverage
5	68% 106/155	63% 36/57

Breakdown by Package

Name	Number of Classes	Line Coverage	Mutation Coverage
br.com.empresa.acoes.io	1	91% 31/34	67% 6/9
br.com.empresa.acoes.model	3	97% 75/77	86% 30/35
br.com.empresa.acoes.ui	1	0% 0/44	0% 0/13

8) Conclusão

A execução desse trabalho permitiu colocar em prática todos os conhecimentos adquiridos em sala de aula e em estudos em casa. Além disso, percebi o quanto um grafo pode auxiliar na hora da criação dos testes unitários. Também pude relembrar alguns conceitos de Programação Orientada a Objetos, algumas nomenclaturas de Java e de ter conhecido novas ferramentas de teste como o JUnit, SonarLint e Pitclipse.