

Disciplina: **Teste**

Antonio Carlos Ferreira de Almeida - RA:2438194

Aberto: segunda, 30 jan 2023, 08:00

Vencimento: domingo, 12 fev 2023, 23:59

Seguindo um ciclo *Test Driven Development* (TDD), desenvolva as classes necessárias (usando a linguagem Java e o *framework* JUnit) para resolver o problema descrito abaixo:

"O participante deve implementar uma calculadora de salário de funcionários. Um funcionário contém nome, e-mail, salário-base e cargo. De acordo com seu cargo, a regra para cálculo do salário líquido é diferente:

- *Caso o cargo seja DESENVOLVEDOR, o funcionário terá desconto de 20% caso o salário seja maior ou igual que 3.000,00, ou apenas 10% caso o salário seja menor que isso.*
- *Caso o cargo seja DBA, o funcionário terá desconto de 25% caso o salário seja maior ou igual que 2.000,00, ou apenas 15% caso o salário seja menor que isso.*
- *Caso o cargo seja TESTADOR, o funcionário terá desconto de 25% caso o salário seja maior ou igual que 2.000,00, ou apenas 15% caso o salário seja menor que isso.*
- *Caso o cargo seja GERENTE, o funcionário terá desconto de 30% caso o salário seja maior ou igual que 5.000,00, ou apenas 20% caso o salário seja menor que isso.*

Exemplos de cálculo do salário:

- *DESENVOLVEDOR com salário-base 5,000.00. Salário final = 4.000,00*
- *GERENTE com salário-base de 2.500,00. Salário final: 2.000,00*
- *TESTADOR com salário de 550.00. Salário final: 467,50*

Entregue o código fonte das classes requisitadas em um arquivo único .pdf (coloque o código necessário para responder os itens usando seu editor de texto favorito e salve em um arquivo .pdf para entrega).

Respostas:

01.

```
4
5 package br.calcSalarFunc;
6
7 import java.math.BigDecimal;
8
9
10 public class CalculadoraDeSalarioDeFuncionarios {
11
12     public void calcularReajuste(Funcionarios funcionario) {
13
14         final int menorque = -1;
15
16         final BigDecimal salarioBaseDesenvolvedor = new BigDecimal("3000.0").setScale(2, RoundingMode.HALF_UP);
17         final BigDecimal salarioBaseDBA = new BigDecimal("2000.0").setScale(2, RoundingMode.HALF_UP);
18         final BigDecimal salarioBaseTestador = new BigDecimal("2000.0").setScale(2, RoundingMode.HALF_UP);
19         final BigDecimal salarioBaseGerente = new BigDecimal("5000.0").setScale(2, RoundingMode.HALF_UP);
20
21         final BigDecimal noventaPercent = new BigDecimal(".9");
22         final BigDecimal oitentaECincoPercent = new BigDecimal(".85");
23         final BigDecimal oitentaPercent = new BigDecimal(".8");
24         final BigDecimal setentaECincoentaPercent = new BigDecimal(".75");
25         final BigDecimal setentaPercent = new BigDecimal(".7");
26
27         switch (funcionario.getCargo()) {
28             case "Desenvolvedor":
29                 if (funcionario.getSalarioBase().compareTo(salarioBaseDesenvolvedor) == menorque) {
30                     funcionario.setSalarioBase(
31                         funcionario.getSalarioBase().multiply(noventaPercent)
32                             .setScale(2, RoundingMode.HALF_UP));
33                 } else {
34                     funcionario.setSalarioBase(
35                         funcionario.getSalarioBase().multiply(oitentaPercent)
36                             .setScale(2, RoundingMode.HALF_UP));
37                 }
38
39                 break;
40             case "DBA":
41                 if (funcionario.getSalarioBase().compareTo(salarioBaseDBA) == menorque) {
42                     funcionario.setSalarioBase(
43                         funcionario.getSalarioBase().multiply(oitentaECincoPercent)
44                             .setScale(2, RoundingMode.HALF_UP));
45                 } else {
46                     funcionario.setSalarioBase(
47                         funcionario.getSalarioBase().multiply(setentaECincoentaPercent)
48                             .setScale(2, RoundingMode.HALF_UP));
49                 }
50                 break;
51             case "Testador":
52                 if (funcionario.getSalarioBase().compareTo(salarioBaseTestador) == menorque) {
53                     funcionario.setSalarioBase(
54                         funcionario.getSalarioBase().multiply(oitentaECincoPercent)
55                             .setScale(2, RoundingMode.HALF_UP));
56                 } else {
57                     funcionario.setSalarioBase(
58                         funcionario.getSalarioBase().multiply(setentaECincoentaPercent)
59                             .setScale(2, RoundingMode.HALF_UP));
60                 }
61                 break;
62             case "Gerente":
63                 if (funcionario.getSalarioBase().compareTo(salarioBaseGerente) == menorque) {
64                     funcionario.setSalarioBase(
65                         funcionario.getSalarioBase().multiply(oitentaPercent)
66                             .setScale(2, RoundingMode.HALF_UP));
67                 } else {
68                     funcionario.setSalarioBase(
69                         funcionario.getSalarioBase().multiply(setentaPercent)
70                             .setScale(2, RoundingMode.HALF_UP));
71                 }
72                 break;
73         }
74     }
75 }
76
77 }
```

2.

```

1 package br.calcSalarFunc;
2
3 import java.math.BigDecimal;
4
5 public class Funcionarios {
6     private String nome;
7     private String email;
8     private BigDecimal salarioBase;
9     private String cargo;
10
11     public Funcionarios(String nome, String email, BigDecimal salarioBase, String cargo) {
12         this.nome = nome;
13         this.email = email;
14         this.salarioBase = salarioBase;
15         this.cargo = cargo;
16     }
17
18     public String getNome() {return nome;}
19
20     public void setNome(String nome) {this.nome = nome;}
21
22     public String getEmail() {return email;}
23
24     public void setEmail(String email) {this.email = email;}
25
26     public BigDecimal getSalarioBase() {return salarioBase;}
27
28     public void setSalarioBase(BigDecimal salarioBase) {this.salarioBase = salarioBase;}
29
30     public String getCargo() {return cargo;}
31
32     public void setCargo(String cargo) {this.cargo = cargo;}
33
34 }

```

3.

```

1 package serviceTest;
2
3 import static org.junit.Assert.assertEquals;
4 import java.math.BigDecimal;
5 import java.math.RoundingMode;
6
7 import org.junit.jupiter.api.Test;
8 import br.calcSalarFunc.CalculadoraDeSalarioDeFuncionários;
9 import br.calcSalarFunc.Funcionarios;
10
11 public class CalcSalarFuncServiceTest {
12
13     CalculadoraDeSalarioDeFuncionários calculadoraDeSalarioDeFuncionários =
14         new CalculadoraDeSalarioDeFuncionários();
15
16     @Test
17     public void casoCargoSejaDesenvolvedorDescontoEmFuncaoDaFaixaSalarialUp() {
18         Funcionarios funcionario = new Funcionarios("ACFA", "acfa@email.com",
19             new BigDecimal("5000.00").setScale(2, RoundingMode.HALF_UP), "Desenvolvedor");
20         calculadoraDeSalarioDeFuncionários.calcularReajuste(funcionario);
21         assertEquals(new BigDecimal("4000.0000"), funcionario.getSalarioBase());
22     }
23
24     @Test
25     public void casoCargoSejaDesenvolvedorDescontoEmFuncaoDaFaixaSalarialDown() {
26
27         Funcionarios funcionario = new Funcionarios("ACFA", "acfa@email.com",
28             new BigDecimal("2500.00").setScale(2, RoundingMode.HALF_UP), "Desenvolvedor");
29         calculadoraDeSalarioDeFuncionários.calcularReajuste(funcionario);
30         assertEquals(new BigDecimal("2250.0000"), funcionario.getSalarioBase());
31     }
32

```

```

33@ @Test
34 public void casoCargoSejaDBADescontoEmFuncaoDaFaixaSalarialUp() {
35     Funcionarios funcionario = new Funcionarios("ACFA", "acfa@email.com",
36         new BigDecimal("3000.00").setScale(2, RoundingMode.HALF_UP), "DBA");
37     calculadoraDeSalarioDeFuncionários.calcularReajuste(funcionario);
38     assertEquals(new BigDecimal("2250.0000"), funcionario.getSalarioBase());
39 }
40
41@ @Test
42 public void casoCargoSejaDBADescontoEmFuncaoDaFaixaSalarialDown() {
43     Funcionarios funcionario = new Funcionarios("ACFA", "acfa@email.com",
44         new BigDecimal("550.00").setScale(2, RoundingMode.HALF_UP), "DBA");
45     calculadoraDeSalarioDeFuncionários.calcularReajuste(funcionario);
46     assertEquals(new BigDecimal("467.5000"), funcionario.getSalarioBase());
47 }
48
49@ @Test
50 public void casoCargoSejaTestadorDescontoEmFuncaoDaFaixaSalarialUp() {
51     Funcionarios funcionario = new Funcionarios("ACFA", "acfa@email.com",
52         new BigDecimal("5000.00").setScale(2, RoundingMode.HALF_UP), "Testador");
53     calculadoraDeSalarioDeFuncionários.calcularReajuste(funcionario);
54     assertEquals(new BigDecimal("3750.0000"), funcionario.getSalarioBase());
55 }
56
57@ @Test
58 public void casoCargoSejaTestadorDescontoEmFuncaoDaFaixaSalarialDown() {
59     Funcionarios funcionario = new Funcionarios("ACFA", "acfa@email.com",
60         new BigDecimal("550.00").setScale(2, RoundingMode.HALF_UP), "Testador");
61     calculadoraDeSalarioDeFuncionários.calcularReajuste(funcionario);
62     assertEquals(new BigDecimal("467.5000"), funcionario.getSalarioBase());
63 }
64
65@ @Test
66 public void casoCargoSejaGerenteDescontoEmFuncaoDaFaixaSalarialUp() {
67     Funcionarios funcionario = new Funcionarios("ACFA", "acfa@email.com",
68         new BigDecimal("5000.00").setScale(2, RoundingMode.HALF_UP), "Gerente");
69     calculadoraDeSalarioDeFuncionários.calcularReajuste(funcionario);
70     assertEquals(new BigDecimal("3500.0000"), funcionario.getSalarioBase());
71 }
72
73@ @Test
74 public void casoCargoSejaGerenteDescontoEmFuncaoDaFaixaSalarialDown() {
75     Funcionarios funcionario = new Funcionarios("ACFA", "acfa@email.com",
76         new BigDecimal("2500.00").setScale(2, RoundingMode.HALF_UP), "Gerente");
77     calculadoraDeSalarioDeFuncionários.calcularReajuste(funcionario);
78     assertEquals(new BigDecimal("2000.0000"), funcionario.getSalarioBase());
79 }

```

Finished after 0,296 seconds

Runs: 8/8 Errors: 0 Failures: 0

> CalcSalarFuncServiceTest [Runner: JUnit] Failure Trace

```

ncionario),
getSalarioBase());

lUp() {
    cfa@email.com",
    Mode.HALF_UP), "DBA");
    ncionario);
    getSalarioBase());

lDown() {
    cfa@email.com",
    ode.HALF_UP), "DBA");
    ncionario);
    etSalarioBase());

```