



**Prezado candidato,**

**Para o desenvolvimento dos algoritmos propostos abaixo, não serão permitidos o uso de bibliotecas de terceiros.**

**A questão 1 deve ser desenvolvida em Java, com a bibliotecas nativas do JEE.**

**As questões 2 e 3 devem ser desenvolvidas em Javascript + JQuery. É permitido usar bibliotecas de terceiros.**

**A questão 4 deve ser desenvolvida com HTML e CSS. É permitido usar bibliotecas de terceiros.**

**Ao finalizar o desafio, os códigos devem ser commitados no GitHub/BitBucket.**

**Favor enviar o endereço público dos mesmos para [cristiane.moraes@duotecnologia.com.br](mailto:cristiane.moraes@duotecnologia.com.br)**

1) Cada funcionário tem um departamento. String  
Cada funcionário tem um salário. BigDecimal  
Cada funcionário tem um cargo. String

Dado uma lista de funcionários, seu salário e departamento e cargo, calcule:

- Custo total por departamento
- Custo total por cargo

O retorno deve ser uma implementação da Interface Calculo.

Exemplo de lista:

```
Funcionario funcionario1 = new Funcionario("Assistente", "Administrativo", new BigDecimal(1000.0));  
Funcionario funcionario2 = new Funcionario("Gerente", "Administrativo", new BigDecimal(7000.70));  
Funcionario funcionario3 = new Funcionario("Diretor", "Administrativo", new BigDecimal(10000.45));  
Funcionario funcionario4 = new Funcionario("Assistente", "Financeiro", new BigDecimal(1300.9));  
Funcionario funcionario5 = new Funcionario("Gerente", "Financeiro", new BigDecimal(7500));  
Funcionario funcionario6 = new Funcionario("Diretor", "Financeiro", new BigDecimal(11000.0));  
Funcionario funcionario7 = new Funcionario("Estagiário", "Jurídico", new BigDecimal(700.4));  
Funcionario funcionario8 = new Funcionario("Assistente", "Jurídico", new BigDecimal(1800.90));  
Funcionario funcionario9 = new Funcionario("Gerente", "Jurídico", new BigDecimal(9500.50));  
Funcionario funcionario10 = new Funcionario("Diretor", "Jurídico", new BigDecimal(13000.0));  
  
List<Funcionario> listaFuncionario = new ArrayList<>();  
listaFuncionario.add(funcionario1);  
listaFuncionario.add(funcionario2);  
listaFuncionario.add(funcionario3);  
listaFuncionario.add(funcionario4);  
listaFuncionario.add(funcionario5);  
listaFuncionario.add(funcionario6);  
listaFuncionario.add(funcionario7);  
listaFuncionario.add(funcionario8);  
listaFuncionario.add(funcionario9);  
listaFuncionario.add(funcionario10);
```

## **Objeto Funcionario**

```
package br.com.projuris;
```

```
import java.math.BigDecimal;
```

```
public class Funcionario {
```

```
    private String cargo;
```

```
    private String departamento;
```

```
    private BigDecimal salario;
```

```
    public Funcionario(String cargo, String departamento, BigDecimal salario) {
```

```
        this.cargo = cargo;
```

```
        this.departamento = departamento;
```

```
        this.salario = salario;
```

```
    }
```

```
    // getters e setters;
```

```
}
```

## **Interface Calculo**

```
package br.com.projuris;
```

```
import java.util.List;
```

```
public interface Calculo {
```

```
    public List<CustoCargo> custoPorCargo(List<Funcionario> funcionarios);
```

```
    public List<CustoDepartamento> custoPorDepartamento(List<Funcionario> funcionarios);
```

```
}
```

### **Objeto CustoCargo**

```
package br.com.projuris;

import java.math.BigDecimal;

public class CustoCargo {
    private String cargo;
    private BigDecimal custo;
    // getters e setters;
}
```

### **Objeto CustoDepartamento**

```
package br.com.projuris;

import java.math.BigDecimal;

public class CustoDepartamento {
    private String departamento;
    private BigDecimal custo;
    // getters e setters
}
```

O nome da classe que implementa a lógica deve ser MyCalculo sob o package br.com.projuris.

**2)** A função adicionaDiv deveria apenas adicionar uma child div para cada div existente em um documento. Toda nova div será decorada pela função decoraDiv.  
Por exemplo:

Estado atual do documento

```
<div id="1">Div 1  
  
    <div id="2">Div 2</div>  
  
</div>
```

O resultado esperado ao executar a função adicionarDiv é:

```
<div id="a">Div A  
  
    <div id="b">Div B  
  
        <div>Nova Div</div>  
  
    </div>  
  
    <div>Nova Div</div>  
  
</div>
```

O problema é que o código atual está entrando em looping eterno.

Segue trecho do código com bug. Corrija o erro:

```
<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Teste Projuris</title>

<script>

    function adicionaDiv() {

        var divList = document.getElementsByTagName("div");

        for (var i = 0; i < divList.length; i++) {

            var novaDiv = document.createElement("div");

            decorarDiv(novaDiv);

            divList[i].appendChild(novaDiv);

        }

    }

    function decorarDiv(div) {

        div.textContent = 'Nova Div';

    }

</script>

</head>

<body>

    <div id="1">Div 1

        <div id="2">Div 2</div>

    </div>

</body>

</html>
```

**3)** Existe uma lista que distingue animais por região e criticidade de ameaça de extinção.

```
<div>
  <ul regioao="Norte">
    <li data-animal="condor negro">Ameaçado de Extinção</li>
    <li data-animal="sapo cururu">Abundante</li>
  </ul>
  <ul regioao="sul">
    <li data-animal="mamute">Extinto</li>
  </ul>
</div>
```

Dada esta lista, crie uma função que recebe a região e o nome do animal para retornar a criticidade de ameaça extinção do mesmo.

Exemplo: se a função receber ('Norte', 'condor negro'), deve retornar 'Ameaçado de Extinção'.

```
function situacaoAnimal(regiao, animal) {
  // Implemente a função aqui
}
```

4) O Designer do produto desenhou uma tela de cadastro. Crie um HTML para codificar esta especificação.  
Segue mockup da tela:

Adicionar Pessoa

Cancelar Cadastro

TIPO ★

Pessoa Física

TRATAMENTO

Tratamento

NOME ★

Informe o nome completo

GÊNERO

Masculino

CPF

000.000.000-00

RG

Informe o RG

NASCIMENTO

Informe a data

ADVOGADO? ★

Não

QUALIFICAÇÃO

Insira mais informações sobre a pessoa

PALAVRAS-CHAVE

Informe as palavras-chave desejadas

E-MAIL

Pessoal

Informe o endereço e-mail

+

PRINCIPAL

TELEFONE

Pessoal

(7) 98833-9999

+

PRINCIPAL

Comercial

(7) 99337-2222

+

PRINCIPAL

ENDEREÇO

Pessoal

Informe o endereço completo

+

PRINCIPAL

Ocultar campos

Salvar

Cancelar

O campo “Tipo” permite os valores “Pessoa física” e “Pessoa Jurídica”.

O campo “Gênero” permite os valores “Masculino” e “Feminino”.

O campo “Advogado” permite os valores “Sim” e “Não”.

Os campos de texto como CPF, Telefone, RG e Nascimento possuem máscaras de input.

Não é necessário implementar ações nos clicks de nenhum botão.

Não é necessário submeter o formulário e nem salvar os dados em banco de dados.

Basta criar a tela nos moldes e estilos previstos pelo desenho.