# Avaliação Técnica – Desenvolvedor

As tarefas para avaliação técnica foram desenvolvidas para que possamos avaliar sua lógica, interpretação de especificações, capacidade e criatividade em resolver problemas.

Na Paradigma, trabalhamos com SQL Server, C#, asp.net, javascript, mas você não precisa usar nenhuma linguagem específica para resolver suas tarefas. Utilize a forma que você entende que irá demonstrar o seu melhor.

Se você possui perfil no github informe ele como parte das suas respostas, assim nós poderemos conhecer um pouco mais sobre você e sua maneira de codar 😉.

## Tarefa 1

Escreva uma consulta usando linguagem SQL para encontrar os colaboradores que tem o salário mais alto em cada um dos departamentos.

A tabela Pessoa possui a relação de todos os colaboradores de uma empresa. Cada pessoa tem um Id, um salário, e também uma coluna para o ID do departamento.

|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **Nome** | **Salario** | **DeptId** |
| 1 | Joe | 70000 | 1 |
| 2 | Henry | 80000 | 2 |
| 3 | Sam | 60000 | 2 |
| 4 | Max | 90000 | 1 |

A tabela Departamento contém a lista de departamentos da empresa.

|  |  |
| --- | --- |
| **Id** | **Nome** |
| 1 | TI |
| 2 | Vendas |

Tarefa 1 - Objetivo: Encontrar o colaborador com o maior salário em cada departamento.

Sql  
  
SELECT

d.Nome AS Departamento,

p.Nome AS Pessoa,

p.Salario

FROM Pessoa p

JOIN Departamento d

ON p.DeptId = d.Id

WHERE p.Salario = (

SELECT MAX(p2.Salario)

FROM Pessoa p2

WHERE p2.DeptId = p.DeptId

);

Resultado esperado:

|  |  |  |
| --- | --- | --- |
| **Departamento** | **Pessoa** | **Salario** |
| TI | Max | 90000 |
| Vendas | Henry | 80000 |

## Tarefa 2

Dado um array inteiro sem duplicidade, construa um algoritmo de uma árvore a partir das seguintes regras:

* A raiz da árvore deve ser o maior valor do array
* Os galhos da esquerda devem ser compostos somente por números à esquerda do valor raiz, na ordem decrescente
* Os galhos da direita devem ser compostos somente por número à direita do valor raiz, na ordem decrescente

Você pode apresentar um algoritmo em pseudo-código ou em qualquer linguagem de programação moderna.

Seu algoritmo deve ser capaz de resolver os dois cenários abaixo:

|  |  |
| --- | --- |
| **Cenario 1**  Array de entrada: [3, 2, 1, 6, 0, 5]  Raiz: 6  Galhos da esquerda: 3, 2, 1  Galhos da direita: 5, 0  6  / \  3 5  / \  2 0  /  1 | **Cenario 2**  Array de entrada: [7, 5, 13, 9, 1, 6, 4]  Raiz: 13  Galhos da esquerda: 7, 5  Galhos da direita: 9, 6, 4, 1  13  / \  7 9  / \  5 6  \  4  \  1 |

Tarefa Objetivo: Construir uma árvore binária a partir de um array seguindo as regras especificadas

javascript

function construirArvore(nums) {

if (!nums.length) return null;

const maxIndex = nums.indexOf(Math.max(...nums));

return {

valor: nums[maxIndex],

esquerda: construirArvore(nums.slice(0, maxIndex)),

direita: construirArvore(nums.slice(maxIndex + 1))

};

}

function imprimirArvore(node, nivel = 0) {

if (!node) return;

console.log(' '.repeat(nivel \* 2) + node.valor);

imprimirArvore(node.esquerda, nivel + 1);

imprimirArvore(node.direita, nivel + 1);

}

console.log("Cenário 1:");

imprimirArvore(construirArvore([3, 2, 1, 6, 0, 5]));

console.log("\nCenário 2:");

imprimirArvore(construirArvore([7, 5, 13, 9, 1, 6, 4]));

**Saída esperada:**

Cenário 1:

6

3

2

1

5

0

Cenário 2:

13

7

5

9

6

4

1