Lista de Exercícios - JavaScript

Questão: Contador Regressivo com Condição

Crie uma função em JavaScript chamada `contagemRegressiva` que recebe dois parâmetros: `inicio` (um número inteiro) e `parada` (um número inteiro).

A função deve realizar uma contagem regressiva, começando do número `inicio` e indo até (mas não incluindo) o número `parada`. Durante a contagem, a função deve imprimir cada número no console.

A contagem deve parar se o número atual se tornar menor que 50. Caso contrário, a contagem continuará normalmente até a condição `parada` ser atingida.

Exemplos de uso:

```
console.log("--- Exemplo 1: Parada antes do final ---");
contagemRegressiva(120, 10);
// Saída esperada:
// 120
// 119
// ...
// 51
// 50
// (A contagem para em 50, pois o próximo número seria menor que 50)

console.log("\n--- Exemplo 2: Contagem completa ---");
contagemRegressiva(45, 10);
// Saída esperada:
// 45
// 44
// ...
// 11
// (A contagem completa, pois o início já é menor que 50)

console.log("\n--- Exemplo 3: Contagem vazia ---");
contagemRegressiva(10, 20);
// Saída esperada:
// (Nenhuma saída, pois a condição do loop não é satisfeita de início)
```

Instruções para Implementação:

Questão: Analisador de Sequência de Votação

Crie uma função em JavaScript chamada `analisarSequenciaDeVotos` que recebe um único parâmetro: `sequencia` (um array de números inteiros). Cada número representa a idade de uma pessoa que votou.

A função deve processar a sequência de votos **enquanto** a idade for válida para voto, seguindo as seguintes regras:

- **Uma idade é considerada válida para voto se for 16 ou superior.**
- O processamento da sequência deve **parar no primeiro voto com idade inválida** (menor que 16).
- Se a sequência inteira consistir em idades válidas, processe-a completamente.

A função deve retornar um **objeto** que contém as seguintes informações sobre os votos válidos processados:

- `totalVotos`: (number) O número total de votos válidos processados.
- `mediaIdade`: (number) A média das idades dos votos válidos.
- `votoMaisJovem`: (number) A idade do votante mais jovem na sequência processada.

Exemplos de uso:

```
console.log(analisarSequenciaDeVotos([25, 40, 18, 55, 15, 60]));
/* Saída esperada:
{
  totalVotos: 4,
  mediaIdade: 34.5,
  votoMaisJovem: 18
}
*/

console.log(analisarSequenciaDeVotos([30, 20, 16, 70, 80]));
/* Saída esperada:
{
  totalVotos: 5,
  mediaIdade: 43.2,
  votoMaisJovem: 16
}
*/

console.log(analisarSequenciaDeVotos([10, 20, 30]));
/* Saída esperada:
{
  totalVotos: 0,
  mediaIdade: 0,
  votoMaisJovem: null
```

}
*/

Instruções para Implementação:

Questão: Validador de Preços de Produtos

Crie uma função em JavaScript chamada `processarProduto` que recebe um único parâmetro: `produto` (um objeto).

O objeto `produto` deve ter as seguintes propriedades:

- `nome`: (string) O nome do produto.
- `preco`: (número) O preço do produto.

A função deve realizar as seguintes verificações:

1. Verificar se o objeto `produto` e suas propriedades `nome` e `preco` existem.
2. Verificar se o `preco` é um número válido e positivo.

Se qualquer uma dessas verificações falhar, a função deve **lançar (throw) um erro** com uma mensagem descritiva.

Se todas as verificações passarem, a função deve retornar uma string no formato: `"O produto [nome] custa R$ [preco]."`

Em seguida, crie uma **segunda função** chamada `validarEExibirProduto` que recebe um objeto `produto` e faz o seguinte:

- Use um bloco **`try`** para tentar chamar a função `processarProduto`.
- Use um bloco **`catch`** para capturar qualquer erro que possa ser lançado.
- No bloco `catch`, imprima a mensagem de erro no console.
- No bloco `try`, se a chamada a `processarProduto` for bem-sucedida, imprima a string de retorno no console.

Exemplos de uso:

// Casos que devem funcionar
validarEExibirProduto({ nome: "Teclado Mecânico", preco: 250.75 }); // Saída: "O produto Teclado Mecânico custa R$ 250.75."
validarEExibirProduto({ nome: "Mouse", preco: 80 }); // Saída: "O produto Mouse custa R$ 80."

// Casos que devem lançar e capturar um erro
validarEExibirProduto({ nome: "Fone de Ouvido", preco: -100 }); // Saída de erro: "Erro: O preço do produto deve ser um número positivo."

validarEExibirProduto({ nome: "Monitor" }); // Saída de erro: "Erro: A propriedade 'preco' está faltando ou não é válida."
validarEExibirProduto({ preco: 50 }); // Saída de erro: "Erro: A propriedade 'nome' está faltando."
validarEExibirProduto(null); // Saída de erro: "Erro: O objeto do produto é nulo ou indefinido."
validarEExibirProduto("string"); // Saída de erro: "Erro: O objeto do produto é nulo ou indefinido."

Instruções para Implementação:

Questão: Sistema de Gestão de Livros

Crie um **construtor** em JavaScript chamado `Livro`. O construtor receber três parâmetros: `titulo`, `autor` e `anoPublicacao`.

Dentro do construtor, você deve:

1. Verificar se o `titulo` e o `autor` não são strings vazias. Se forem, lance um `Error` com uma mensagem descritiva (ex: "Título do livro não pode ser vazio.").
2. Verificar se o `anoPublicacao` é um número e se está entre 1500 e o ano atual (inclusive). Se não for, lance um `Error` com uma mensagem descritiva (ex: "Ano de publicação inválido.").
3. Se todas as verificações passarem, atribua os valores dos parâmetros às propriedades do objeto (`this.titulo`, `this.autor`, `this.anoPublicacao`).

Por fim, faça exemplos válidos e inválidos.