UNIVERSIDADE FEDERAL DO CEARÁ

Questões

1) Você foi contratado por uma empresa de desenvolvimento web, porém a empresa queria "reinventar a roda" e decidiu que todo o arquivo HTML devia ser escrito em javascript, você com o seus conhecimentos de closures decidiu implementar uma função chamada addHTMLTag que utilizaria o conceito de closure para gerar várias novas funções que permitissem a você, desenvolvedor, passar uma string qualquer e circundar essa string com uma tag HTML. Seu código final ficou assim:

```
let addPTag = addHTMLTag("p")
let addH1Tag = addHTMLTag("h1")
let addH2Tag = addHTMLTag("h1")

console.log(addH1Tag("Esse é meu título clickbait"))
console.log(addH2Tag("Um subtítulo para te deixar curioso"))
console.log(addPTag("Um texto TLDR -too long, don't read-"))
```

O resultado desse código foi esse:

```
<h1> Esse é meu título </h1> <h2> Um subtítulo para te deixar curioso </h2>
<p> Um texto TLDR -too long, don't read- </p>
```

O seu trabalho é implementar essa funcionalidade usando closures

2) Você foi desafiado a desenvolver uma função em JavaScript que verifica se uma palavra é um palíndromo. Um palíndromo é uma palavra que permanece a mesma, mesmo se lida de trás para frente. Por exemplo, "arara" é um palíndromo, pois, ao inverter a ordem das letras, obtemos a mesma palavra. Sua tarefa é implementar a função isPalindrome, utilizando recursão, que recebe uma string como parâmetro e retorna true se a palavra for um palíndromo e false caso contrário.

**Exemplos de Palavras Palíndromas: aibofobia, arara, esse, osso, ovo, radar, reviver

Exemplo de entrada e saída esperadas:

Entrada:

"arara"

Saída:

true

3) Implemente uma função chamada smdSort que recebe como parâmetro uma lista de números inteiros e retorna a lista ordenada em ordem crescente. O algoritmo de ordenação SMD funciona da seguinte maneira:

Inicialmente, considera-se que o primeiro elemento da lista está ordenado. Em seguida, percorre-se a lista a partir do segundo elemento. Para cada elemento atual, o algoritmo realiza a inserção do elemento na posição correta da parte já ordenada da lista. Para fazer a inserção, compara-se o elemento atual com os elementos anteriores na parte ordenada e move-se os elementos maiores uma posição para a direita, a fim de abrir espaço para o elemento atual. Repete-se o processo até que todos os elementos da lista estejam ordenados.

Sua tarefa é implementar a função smdSort seguindo as instruções acima. A função deve retornar a lista original, porém ordenada.

Exemplo de entrada e saída esperadas:

Entrada:

[5, 3, 8, 2, 1]

Saída:

[1, 2, 3, 5, 8]

4) Implemente um passo a passo, sem código (no máximo pseudocódigo) do algorítimo de ordenação mergeSort

5) Implemente o algoritmo Bubble Sort em JavaScript para ordenar o estoque da loja de produtos eletrônicos. O estoque é representado por um array de objetos, onde cada objeto contém informações sobre um produto, incluindo o código único de identificação. O Bubble Sort é um algoritmo de ordenação simples que percorre repetidamente a lista, comparando pares de elementos adjacentes e os trocando de posição se estiverem na ordem errada.

   Exemplo de entrada e saída esperadas:

   Entrada:

```
const estoque = [
{ codigo: 102, nome: 'Celular', quantidade: 5 },
{ codigo: 305, nome: 'Notebook', quantidade: 3 },
{ codigo: 201, nome: 'Tablet', quantidade: 2 },
{ codigo: 415, nome: 'Fones de Ouvido', quantidade: 10 },
];
```

   Saída:

```
{ codigo: 102, nome: 'Celular', quantidade: 5 },
{ codigo: 201, nome: 'Tablet', quantidade: 2 },
{ codigo: 305, nome: 'Notebook', quantidade: 3 },
{ codigo: 415, nome: 'Fones de Ouvido', quantidade: 10 },
```

   Escreva uma função chamada ordenarEstoque que receba o array de objetos do estoque como entrada e retorne o array ordenado em ordem crescente de acordo com o código dos produtos. Certifique-se de testar sua implementação com diferentes conjuntos de dados para garantir sua corretude e eficiência.

   Dica: Você pode utilizar a propriedade codigo de cada objeto do estoque para realizar as comparações no algoritmo Bubble Sort.

Dicas

Não lembra como funciona closures? Toma esse link, lê até o fim e tenta fazer a questão Basicamente vc vai criar uma função que vai retornar uma outra função ao ser usada, essa outra função vai lembrar do que vc armazenou dentro da função mais externa. Após ler, tente resolver a primeira. Closures

Em Recursão, sempre precisamos entender o caso base, pois é nele que nós vamos conseguir sair da recursão, vou dar uma ajuda a vcs, que tal comparar o primeiro e o último elemento da string? ajuda2: e se eu ler “a” de trás pra frente eu tenho um palíndromo? Ficam ai as dicas!

Veja por si só o funcionamento do smdSort (Ele tem outro nome na vida real) smdSort

Recomendações

Para as duas últimas questões sugiro que não usem o chatGPT, mas peguem um papel e desenhem o que vocês querem fazer, ou desenhem o passo a passo da questão, e tente, por meio do seu desenho, visualizar o que vai acontecer.

No mais, me procurem no Discord ou nas terças-feiras