Atividade 1: Criando uma Lista de Tarefas Interativa

Problema: Você deve criar uma aplicação web simples que permite aos usuários adicionar, remover e marcar tarefas como concluídas em uma lista de tarefas. Para isso, você precisará manipular o DOM para atualizar a interface do usuário.

Instruções: Crie um arquivo HTML básico com a estrutura inicial, incluindo um campo de entrada de texto para adicionar tarefas, um botão "Adicionar Tarefa" e uma lista onde as tarefas serão exibidas.

```
<!DOCTYPE html>
<html>
<head>
  <title>Lista de Tarefas</title>
</head>
<body>
  <h1>Lista de Tarefas</h1>
  <input type="text" id="tarefa" placeholder="Adicione uma tarefa">
  <button id="adicionar">Adicionar Tarefa</button>
  <ul id="lista-tarefas">
    <!-- As tarefas serão adicionadas aqui -->
  </ul>
  <script src="script.js"></script>
</body>
</html>
```

1. Crie um arquivo JavaScript chamado "script.js" e inicie-o com o seguinte código:

```
document.addEventListener('DOMContentLoaded', function () {
  // Seu código aqui
});
```

a. Dentro do evento de carregamento do DOM, adicione o código JavaScript para manipular os elementos HTML e criar a funcionalidade da lista de tarefas:
   - Quando o botão "Adicionar Tarefa" for clicado, pegue o texto do campo de entrada e adicione-o como uma nova tarefa na lista.
   - Crie um botão de exclusão para cada tarefa na lista que permita aos usuários remover a tarefa.
   - Adicione a capacidade de marcar tarefas como concluídas ou não concluídas quando clicadas.

b. Use classes CSS para estilizar as tarefas concluídas de forma diferente das tarefas não concluídas
c. Teste sua aplicação interativa de lista de tarefas para garantir que os elementos do DOM estejam sendo manipulados corretamente.

Atividade 2: Manipulação de Arrays e Objetos

Problema: Imagine que você está desenvolvendo um sistema de gerenciamento de estoque. Você recebeu um array de objetos que representam produtos em estoque. Cada objeto possui propriedades como nome, preço e quantidade em estoque.

Instruções: Sua tarefa é escrever uma função que calcule o valor total do estoque.

```
const produtos = [
```

```
  { nome: 'Laptop', preco: 1000, quantidade: 5 },
  { nome: 'Mouse', preco: 20, quantidade: 10 },
  { nome: 'Teclado', preco: 30, quantidade: 8 }
];

function calcularValorTotalEstoque(produtos) {
  // Sua implementação aqui
}

const valorTotal = calcularValorTotalEstoque(produtos);

console.log('Valor total do estoque:', valorTotal);
```

Atividade 3: Criando um Jogo da Forca

Passo 1: HTML

1. Crie um arquivo HTML chamado index.html.
2. Dentro do arquivo HTML, crie a estrutura básica, incluindo as tags <html>, <head> e <body>.
3. No <head>, adicione um título para o jogo, por exemplo, "Jogo da Forca".
4. No <body>, crie uma área para exibir a palavra oculta e os espaços vazios para as letras a serem adivinhadas.
5. Adicione um campo de entrada para que o jogador insira uma letra.
6. Crie um botão "Adivinhar" para o jogador submeter sua tentativa.
7. Adicione uma área para exibir o número de tentativas restantes e as letras erradas já adivinhadas.

Exemplo de estrutura HTML básica:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,initial-scale=1.0">
    <title>Jogo da Forca</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <h1>Jogo da Forca</h1>
    <div id="palavra-oculta">_ _ _ _ _ _</div>
    <input type="text" id="letra">
    <button id="adivinhar">Adivinhar</button>
    <div id="tentativas">Tentativas restantes: 6</div>
    <div id="letras-erradas">Letras erradas: </div>

    <script src="script.js"></script>
</body>
</html>
```

Passo 2: CSS

1. Crie um arquivo CSS chamado styles.css.
2. Estilize os elementos HTML para tornar o jogo mais atraente e legível.
3. Defina o tamanho, a cor e a fonte do texto.
4. Adicione estilos ao campo de entrada, botão e outras partes do jogo para torná-lo visualmente agradável e intuitivo.

Exemplo de estilos CSS básicos:

```css
body {
    font-family: Arial, sans-serif;
    text-align: center;
}

h1 {
    color: #333;
}

#palavra-oculta {
    font-size: 24px;
    margin: 20px 0;
}

#letra {
    padding: 5px;
    font-size: 16px;
    margin-bottom: 10px;
}

#tentativas {
    margin-top: 20px;
    font-weight: bold;
}

#letras-erradas {
    color: red;
}
```

Passo 3: JavaScript

1. Crie um arquivo JavaScript chamado script.js.
2. Defina uma lista de palavras para o jogo.
3. Escolha uma palavra aleatória da lista para ser a palavra oculta.
4. Implemente a lógica do jogo: verificar se a letra adivinhada está na palavra, atualizar a exibição da palavra oculta, verificar se o jogador ganhou ou perdeu, etc.
5. Atualize a exibição das tentativas restantes e das letras erradas a cada tentativa do jogador.

Exemplo de código JavaScript básico:

```javascript
const palavras = ["javascript", "html", "css", "programacao", "computador"];
let palavraOculta = // sorteie uma palavra
let letrasErradas = [];
let tentativasRestantes = 6;

document.getElementById("adivinhar").addEventListener("click", function() {
    const letra = document.getElementById("letra").value.toLowerCase();
    if (letra.length !== 1 || !/^[a-zA-Z]+$/.test(letra)) {
        alert("Por favor, insira uma letra válida.");
        return;
    }
    if (palavraOculta.includes(letra)) {
        // Atualizar exibição da palavra oculta
    } else {
        letrasErradas.push(letra);
        tentativasRestantes--;
        // Atualizar exibição das letras erradas e tentativas restantes
    }
```

```
    // Verificar se o jogador ganhou ou perdeu
});
```