INSTITUTO SUPERIOR POLITÉCNICO DE TETE

EXERCICIOS DE JAVASCRIPT

Introdução:

Escolha um dos exercícios e faça o que se pedi.

Instruções de Submissão:

Depois de concluírem o exercício, por favor, submetam o seu código na plataforma Moodle.

Prazo de Submissão:

Por favor, submetam o seu trabalho até o final da semana para que possamos fornecer feedback em tempo hábil.

NB: Escolha só 1 exercício.

1. **Exercício Prático: Calculadora Simples**

Descrição:

Desenvolva uma calculadora simples que permita ao usuário realizar operações básicas de matemática, como adição, subtração, multiplicação e divisão.

Requisitos:

1. **Interface de Usuário:**
   - Crie uma interface de usuário intuitiva que exiba os botões numéricos (0-9) e os botões de operação (+, -, *, /).
   - Mostre a entrada atual do usuário e o resultado da operação na tela.
2. **Operações Básicas:**
   - Implemente as operações de adição, subtração, multiplicação e divisão.
   - Garanta que as operações sejam realizadas corretamente e que os resultados sejam exibidos na tela.
3. **Funcionalidades Adicionais:**

- Permita que o usuário limpe a entrada atual ou o resultado para começar uma nova operação.
- Adicione suporte para cálculos sequenciais, ou seja, o resultado de uma operação pode ser usado como entrada para a próxima.

4. **Teste de Funcionalidade:**
    - Teste a calculadora com uma variedade de operações para garantir que os resultados sejam precisos e que a interface do usuário seja responsiva.

Dicas:

- Utilize HTML, CSS e JavaScript para construir a interface de usuário e implementar a lógica da calculadora.
- Divida o código em funções separadas para lidar com diferentes aspectos da aplicação, como a entrada do usuário, as operações matemáticas e a atualização da interface.
- Experimente adicionar recursos extras, como histórico de cálculos ou suporte para operações matemáticas mais avançadas, como exponenciação ou raiz quadrada.

2. **Exercício Prático: Lista de Tarefas**

Descrição:

Desenvolva uma aplicação de lista de tarefas onde os usuários possam adicionar, remover e marcar tarefas como concluídas.

Requisitos:

1. **Interface de Usuário:**
    - Crie uma interface de usuário que permita aos usuários visualizar as tarefas existentes, adicionar novas tarefas, marcar tarefas como concluídas e remover tarefas da lista.
    - Mostre as tarefas em uma lista na tela, com opções para marcar como concluídas e excluir.
2. **Adição de Tarefas:**
    - Permita que os usuários adicionem novas tarefas digitando o texto da tarefa em um campo de entrada e pressionando Enter ou clicando em um botão "Adicionar".
3. **Marcação de Tarefas Concluídas:**
    - Ao clicar em uma tarefa na lista, ela deve ser marcada como concluída e ter sua aparência modificada para indicar isso (por exemplo, atravessar o texto ou alterar a cor de fundo).
4. **Remoção de Tarefas:**

- Permita que os usuários removam uma tarefa da lista clicando em um botão "Excluir" ao lado dela.

5. **Persistência de Dados:**
    - Salve as tarefas adicionadas pelo usuário para que elas persistam entre as sessões. Você pode usar localStorage ou IndexedDB para isso.
6. **Design Responsivo:**
    - Certifique-se de que a aplicação seja responsiva e tenha uma boa aparência em dispositivos móveis e desktop.

Dicas:

- Use HTML, CSS e JavaScript para construir a interface de usuário e implementar a lógica da lista de tarefas.
- Divida o código em funções separadas para lidar com diferentes funcionalidades, como adição, marcação e remoção de tarefas.
- Experimente adicionar recursos extras, como filtragem de tarefas por status (concluídas / não concluídas), ordenação por data de criação ou prioridade, e edição de tarefas.

3. <u>Exercício Prático: Conversor de Unidades</u>

Descrição:

Desenvolva um conversor de unidades que converta unidades comuns, como Celsius para Fahrenheit, metros para pés, quilogramas para libras, etc.

Requisitos:

1. **Interface de Usuário:**
    - Crie uma interface de usuário que permita aos usuários selecionar as unidades de entrada e saída, inserir um valor e ver o resultado da conversão.
    - Forneça opções para diferentes tipos de conversão, como temperatura, comprimento, peso, etc.
2. **Conversão de Unidades:**
    - Implemente as fórmulas de conversão para diferentes tipos de unidades. Por exemplo:
        - Para converter Celsius para Fahrenheit: **F = C × 9/5 + 32**
        - Para converter metros para pés: **ft = m × 3.28084**
        - Para converter quilogramas para libras: **lb = kg × 2.20462**
3. **Seleção de Unidades:**

- Permita que os usuários escolham as unidades de entrada e saída de uma lista suspensa ou botões de opção.

4. **Exibição de Resultados:**
    - Mostre o resultado da conversão de forma clara e legível para o usuário, preferencialmente com precisão adequada (por exemplo, duas casas decimais para temperatura).
5. **Design Responsivo:**
    - Certifique-se de que a aplicação seja responsiva e tenha uma boa aparência em dispositivos móveis e desktop.

Dicas:

- Use HTML, CSS e JavaScript para construir a interface de usuário e implementar a lógica de conversão de unidades.
- Organize as diferentes fórmulas de conversão em funções separadas para facilitar a manutenção e reutilização do código.
- Considere adicionar opções para conversão bidirecional (ou seja, converter de A para B e de B para A) e unidades adicionais para expandir a funcionalidade do conversor.

4. Exercício Prático: Galeria de Imagens

Descrição:

Desenvolva uma galeria de imagens onde o usuário pode navegar entre diferentes imagens e ver uma versão ampliada de cada uma.

Requisitos:

1. **Imagens:**
    - Selecione um conjunto de imagens para exibir na galeria. As imagens podem ser relacionadas a um tema específico ou variadas.
2. **Interface de Usuário:**
    - Crie uma interface de usuário que exiba as miniaturas das imagens, permitindo que o usuário clique em uma para vê-la em tamanho ampliado.
    - Adicione botões de navegação para permitir que o usuário passe para a próxima ou anterior imagem na galeria.
3. **Ampliação de Imagens:**

- Ao clicar em uma miniatura, exiba a imagem em tamanho ampliado em uma área destacada da tela, com a capacidade de fechar a imagem ampliada e voltar à visualização da miniatura.

4. **Funcionalidades Adicionais (opcional):**
    - Implemente um efeito de carrossel para transição suave entre as imagens.
    - Adicione controles de zoom para permitir que o usuário amplie ou reduza a imagem ampliada.
    - Integre uma funcionalidade de pesquisa para permitir que o usuário filtre as imagens por palavra-chave ou categoria.

Dicas:

- Use HTML, CSS e JavaScript para construir a interface de usuário e implementar a funcionalidade da galeria de imagens.
- Utilize eventos de clique para capturar as interações do usuário com as miniaturas e implementar a exibição das imagens ampliadas.
- Organize as imagens em uma estrutura de dados adequada, como um array de objetos com informações sobre cada imagem (URL, título, descrição, etc.).
- Teste a galeria com diferentes conjuntos de imagens para garantir que ela funcione corretamente em todos os casos.

5. **Exercício Prático: Validador de Formulário**

Descrição:

Crie um formulário com campos obrigatórios e, opcionalmente, campos com validação específica, como e-mail ou número de telefone.

Requisitos:

1. **Campos Obrigatórios:**
    - Defina os campos obrigatórios no formulário, como nome, e-mail, senha, etc.
    - Garanta que o usuário preencha esses campos antes de enviar o formulário.
2. **Validação de Campos Específicos:**
    - Adicione validação específica para campos como e-mail, número de telefone, data de nascimento, etc.
    - Use expressões regulares ou bibliotecas de validação para garantir que os dados inseridos nos campos correspondam aos formatos desejados.
3. **Feedback de Validação:**

- Forneça feedback claro e informativo para o usuário quando um campo obrigatório não for preenchido ou quando um campo for preenchido incorretamente.
- Exiba mensagens de erro ao lado dos campos com problemas e destaque esses campos para chamar a atenção do usuário.

4. **Envio do Formulário:**
    - Permita que o usuário envie o formulário somente quando todos os campos obrigatórios estiverem preenchidos e validados com sucesso.
    - Implemente a lógica de envio do formulário para processar os dados inseridos pelo usuário.
5. **Funcionalidades Adicionais (opcional):**
    - Adicione uma barra de progresso para indicar o status de preenchimento do formulário.
    - Implemente validação em tempo real à medida que o usuário preenche os campos, em vez de apenas ao enviar o formulário.

Dicas:

- Use HTML para construir a estrutura do formulário e adicionar atributos como **required** para campos obrigatórios e **pattern** para campos com validação específica.
- Use CSS para estilizar o formulário e fornecer feedback visual aos usuários sobre o status de validação dos campos.
- Use JavaScript para adicionar lógica de validação personalizada e manipular o envio do formulário.
- Teste o formulário com diferentes cenários para garantir que todas as validações estejam funcionando corretamente.

6. Exercício Prático: Gerador de Senhas

Descrição:

Desenvolva uma aplicação que gere senhas aleatórias com base em critérios especificados pelo usuário, como comprimento e tipos de caracteres permitidos.

Requisitos:

1. **Interface de Usuário:**
    - Crie uma interface de usuário que permita ao usuário especificar o comprimento da senha desejada e escolher quais tipos de caracteres incluir na senha (letras maiúsculas, minúsculas, números, caracteres especiais, etc.).

- Forneça um botão ou uma opção para gerar uma nova senha com base nos critérios especificados.

2. **Geração de Senha:**
    - Implemente a lógica para gerar uma senha aleatória com base nos critérios especificados pelo usuário.
    - Use caracteres aleatórios dos tipos selecionados pelo usuário para construir a senha.
3. **Exibição da Senha:**
    - Exiba a senha gerada na interface de usuário para que o usuário possa copiá-la e usá-la conforme necessário.
4. **Funcionalidades Adicionais (opcional):**
    - Adicione opções adicionais para permitir que o usuário especifique outros critérios para a senha, como exigir pelo menos um caractere de cada tipo selecionado ou evitar caracteres repetidos.
    - Implemente uma funcionalidade para copiar a senha para a área de transferência do usuário com um único clique.

Dicas:

- Use HTML, CSS e JavaScript para construir a interface de usuário e implementar a lógica de geração de senha.
- Use métodos e propriedades do objeto **Math** do JavaScript para gerar números aleatórios e selecionar caracteres aleatórios.
- Utilize expressões regulares para validar os critérios especificados pelo usuário e garantir que a senha gerada atenda a esses critérios.
- Teste o gerador de senha com diferentes combinações de critérios para garantir que ele produza senhas seguras e variadas.