UTILIZANDO IF ELSE

Vamos pensar em um sistema que possui controle de permissão. Um diretor tem acesso total, um gerente acesso regional e um vendedor acesso local. Faça uma tela onde aparece a mensagem do tipo de acesso de acordo com o cargo de cada pessoa, caso o cargo não esteja mencionado acima informar que o usuário não foi reconhecido, por tanto sem acesso.

ESSA VAI DE BRINDE PARA VOCÊS

CRIEM UM ARQUIVO DO TIPO .JS EXEMPLO:

excontrole01.js

```
let permissao
permissao = 'caixa'


if (permissao == 'vendedor') {
    console.log('acesso local')
} else if (permissao == 'gerente') {
    console.log('acesso regional')
} else if (permissao == 'diretor') {
    console.log('acesso total')
} else {
    console.log('Usuário não
reconhecido!')
}
```

Só para garantir que você estão conseguindo encaixar na cabeça de o casamento de html com JavaScript, repliquem esse mesmo exemplo em um arquivo do tipo .html exemplo: excontrole01.html

Com um acréscimo do usuário poder inserir qual a função que ele pertence

```
<body>
    <script>

let permissao = prompt("digite sua funcao")

if (permissao == 'vendedor') {
   window.alert('acesso local')
} else if (permissao == 'gerente') {
    window.alert('acesso regional')
} else if (permissao == 'diretor') {
    window.alert('acesso total')
} else {
    window.alert('Usuário não reconhecido!')
}
    </script>
</body>
</html>
```

ESTRUTURA DE CONTROLE SWITCH | CASE

PROFª LARISSA RIBEIRO

DEFINIÇÃO

O switch case é uma estrutura de controle em JavaScript que permite executar diferentes blocos de códigos com base em diferentes valores.

É uma maneira consistente e eficiente de lidar com diversos valor

1. Se a condição for correspondida, o programa executa as instruções associadas.
2. Se múltiplos casos corresponderem ao valor, o primeiro caso que corresponder é selecionado, mesmo se os casos não forem iguais entre si.
3. O programa primeiro procura por um caso cuja expressão avalie como tendo o mesmo valor que o input da expressão (usando a comparação de igualdade estrita, ==), transferindo assim o controle para a cláusula encontrada e, em seguida, executando as instruções associadas.

4. Caso nenhum caso seja correspondido, o programa procura pela cláusula opcional default, que, se encontrada, tem o controle transferido a ela, executando suas instruções associadas.
5. Por convenção, a cláusula default é a última, mas não é algo obrigatório.
6. A instrução opcional break associada a cada case garante que o programa saia da condicional switch assim que a instrução correspondente for executada e continue a execução após o switch.

SINTAXE

Padrão de escrita

```
switch (expressão) {
  case valor1:
    // Instruções executadas quando o resultado da expressão for igual a valor1
    break;
  case valor2:
    // Instruções executadas quando o resultado da expressão for igual a valor2
    break;
  // ...
  case valueN:
    // Instruções executadas quando o resultado da expressão for igual a valorN
    break;
  default:
    // Instruções executadas quando o valor da expressão é diferente de todos os cases
    break;
}
```

E se esquecermos de utilizar o break

O switch é uma condicional que avalia uma expressão e combina o valor dessa expressão com cláusulas case. Cada case contém instruções associadas. Quando a expressão corresponde a um case, as instruções desse case são executadas. No entanto, se omitirmos o break após as instruções de um case, algo interessante acontece: o programa continua a executar as instruções dos case subsequentes, mesmo que eles não correspondam à expressão original.

VAMOS REFAZER O EXERCICIO ANTERIOR

Faça um documento novo do tipo .js

Exemplo:

excontrole03.js

```
let permissao; //vendedor, gerente; diretor
permissao = 'vendedor';
switch (permissao) {
    case 'comum':
        console.log('usuário local')
        break;
    case 'gerente':
        console.log('usuário regional')
        break;
    case 'diretor':
        console.log('usuário total')
        break;
    default:
        console.log('Usuario não reconhecido!')
}
```

O uso do SWITCH CASE m **JavaScript** oferece várias vantagens em comparação com outras estruturas de controle, como condicionais IF-ELSE. Vamos ver essas vantagens:

1. **Clareza e Legibilidade**
    - **O SWITCH CASE** permite agrupar várias opções relacionadas em um único bloco, tornando o código mais organizado e fácil de entender.
    - É especialmente útil quando você precisa lidar com muitos casos diferentes.
2. **Eficiência**
    - **O SWITCH CASE é mais eficiente do que uma série de declações IF- ELSE IF.**
    - Ele avalia a expressão uma vez e, em seguida, direciona o fluxo de controle para o caso correspondente, sem verificar todas as condições subsequentes.

3. **Comparação Estrita**
    - **O SWITCH CASE** usa comparação de igualdade estrita (==), o que significa que ele compara tanto o valor quanto o tipo.
    - Isso evita erros comuns, como comparações acidentais entre tipos diferentes.
4. Melhor Semântica
    - O switch case é semanticamente apropriado para situações em que você está escolhendo entre várias opções com base em um valor específico.
    - Ele reflete a intenção do código de forma mais clara do que uma série de if-else.
5. Tratamento de Casos Especiais:
    - Você pode usar o default para tratar casos não previstos explicitamente.
    - Isso ajuda a evitar erros silenciosos quando nenhum caso corresponde.

6. Agrupamento de Casos
    - Você pode agrupar vários casos para executar o mesmo bloco de código.
    - Por exemplo, case restrições de acesso, no exemplo anterior compartilham o mesmo bloco de código.

Em resumo, o switch case é uma ferramenta poderosa para tomar decisões com base em valores específicos e é especialmente útil quando você tem muitas opções a considerar. No entanto, lembre-se de usá-lo com moderação e sempre considere a clareza e a legibilidade do seu código.

VAMOS TESTAR ESSE EXEMPLO

Faça um documento novo do tipo .js

Exemplo:

excontrole03.js

```
const fruta = "Bananas";
switch (fruta) {
  case "Laranjas":
    console.log("As laranjas custam $0.59 o quilo.");
    break;
  case "Maçãs":
    console.log("Maçãs custam $0.32 o quilo.");
    break;
  case "Bananas":
    console.log("Bananas custam $0.48 o quilo.");
    break;
  case "Cerejas":
    console.log("Cerejas custam $3.00 o quilo.");
    break;
  case "Mangas":
  case "Mamões":
    console.log("Mangas e mamões custam $2.79 o quilo.");
    break;
  default:
    console.log("Fruta não encontrada.");
}
```

Agora faça o exercício anterior dentro de um html e deixe que o usuário solicite a fruta que deseja saber o valor

1. Elabore um algoritmo que leia dois valores do usuário e a operação que ele deseja executar (Operações: soma, subtração, divisão, multiplicação). Execute a operação desejada e imprima na tela.

2. Uma loja fornece 10% de desconto para funcionários e 5% de desconto para clientes vips. Faça um programa que calcule o valor total a ser pago por uma pessoa. O programa deverá ler o valor total da compra efetuada e um código que identifique se o comprador é um cliente comum (1), funcionário (2) ou vip (3).

3. Suponha que estamos construindo um aplicativo de gerenciamento de pedidos de uma lanchonete e precisamos consultar o preço com base no tipo de comida selecionado.
   - Hamburguer R$10,00
   - Batata Frita R$5,00
   - Refrigerante R$2,50
   - Sorvete R$3,00

   Caso o item não esteja na lista informar que não

EXERCICIOS DE FIXAÇÃO, ENTREGAR CÓDIGOS VIA SUAP