Resumo Teórico e Prático de JavaScript

Strict Mode

O Strict Mode no JavaScript é uma maneira de adotar práticas mais seguras e evitar erros comuns. Ele impõe restrições à linguagem, tornando o código mais seguro e eficiente. Para ativá-lo, basta incluir a linha

'"use strict";' no início de um script ou função.

Prático:

```
function exemploStrict() {
    "use strict";
    // Código que segue as regras do strict mode
    let obj = {prop: "valor"};
    obj = "novo valor"; // Erro no strict mode: atribuição incorreta
}
```

console.log para Debug

O console.log é uma ferramenta importante para o debug de código. Ele permite a exibição de valores

no console para acompanhamento do fluxo de execução ou análise de variáveis.

Prático:

```
let a = 5;
let b = 10;
console.log('O valor de a é: ' + a);
console.log('O valor de b é: ' + b);
```

Tratamento de Input por Função

A validação de inputs pode ser feita utilizando funções, como para garantir que o valor do input seja correto
antes de usá-lo.

Prático:

```
function validarNumero(input) {
    if (isNaN(input)) {
        console.log('Entrada inválida, não é um número.');
    } else {
        console.log('Entrada válida: ' + input);
    }
}
validarNumero('abc'); // Saída: Entrada inválida, não é um número.
```

Exception, Try, Catch e Finally

O JavaScript oferece tratamento de exceções com o uso de try, catch e finally.

- O bloco try contém o código que pode lançar uma exceção.
- O bloco catch captura a exceção e permite tratá-la.
- O bloco finally é executado após a execução do try/catch, independentemente de erro.

Prático:

```
try {
    let x = 10;
    let y = 0;
    if (y === 0) {
        throw "Divisão por zero!";
    }
```

```
    let z = x / y;
} catch (e) {
    console.log("Erro: " + e);
} finally {
    console.log("Bloco finally executado.");
}
```

Assertion

Assertions são verificações que validam se uma condição é verdadeira durante a execução do programa.

Se a condição falhar, uma exceção é gerada.

Prático:

```
function assert(valor, mensagem) {
    if (!valor) {
        throw new Error(mensagem);
    }
}
assert(2 > 3, "A condição falhou!"); // Lança erro: A condição falhou!
```