VARIÁVEIS E TIPOS DE DADOS

Variável: Espaço de memória no computador onde se pode armazenar tipos de dados.

JavaScript: 3 tipos de dados

Var, Let

Let só sera acessada nos blocos de códigos no qual foi declarada, blocos determinados por {}

Palavras reservadas (ex: var e let) não podem ser usadas como nomes de variáveis ou funções.

Temos algumas regras para criar nome de variaveis:

- Não pode começar com número
- Não pode ter caracteres especiais
- Não pode utilizar acentos
- Não pode utilizar espaços
- Deve começar com letra minuscula e seguir o padrão camelCase, onde cada palavra é iniciada com maiúsculas e unidas sem espaços.

Atribuição de valores: Para salvar um valor em uma variável, usamos o sinal = e, em seguida, o valor que queremos armazenar.

Em Javascript, o sinal ";" é opcional. Mas é importante que você utilize para o computador conseguir entender onde termina o comando.

String: Cadeia de caracteres que representa qualquer combinação de letras, números e/ou símbolos. **Entre aspas.**

Concatenar strings: símbolo **+**

IMRPIMIR NA TELA: console.log();

Booleano: Representa valores lógicos

True, false

Comparação com valores diferentes

FUNÇÕES

Uma função é um **bloco de código** que podemos invocar quantas vezes forem necessárias.
Pode realizar uma **tarefa específica** e **retornar** um valor.
Nos permite **agrupar** o **código** que vamos **utilizar muitas vezes.**

Funções declaradas e
funções expressadas: são atribuídos como um valor a uma variável.

Estrutura Básica: Palavra reservada function
Nome: definir o nome da função para utilizar quando quiser invocá-la.
Parâmetros: entre (), separa-se com vírgulas, se não houver parâmetros apenas ()
Corpo: código da função é escrito entre {}
Retorno: palabra return seguida do que se quer devolver
Math.abs () retorna o valor absoluto do número que passamos para ele como *parâmetro*.

Math.round () arredonda um número para cima até o número inteiro mais próximo e Math.floor () arredonda um número para baixo até o número inteiro mais próximo.

Math.max () pega dois parâmetros e retorna o maior número, enquanto Math.min () pega dois parâmetros e retorna o menor.

Math.random (). Essa função gera um número **aleatório** decimal entre 0 e 1, e é a base para muitos cálculos usados na programação.

Invocar (executar) uma função é escrevendo o seu nome seguido de abrir e fechar parênteses, respeitando o número e a ordem dos paramêtros.

Os **parâmetros** são as **variáveis** que escrevemos quando **definimos** a função.

Os **argumentos** são os **valores** que enviamos quando **invocamos** a função

CONDICIONAIS

Operadores Lógicos: Permite manipular valores das variáveis, realizar operações e comparar seus valores.

Permite comparar booleanos, devolvendo booleano.

AND && todos os valores devem retornar true

OR || pelo menos um valor true

NOT ! nega a condição

Condicionais: avalia as condições e realiza diferentes ações de acordo com o resultado

IF, ELSE, ELSE IF

ESTRUTURA DE UM CICLO FOR

É composto por **3 partes** que definimos dentro dos parênteses. Juntos, eles nos permitem determinar como as **repetições** serão realizadas, e definir as **instruções** que queremos realizar em cada uma delas.

```
for (início; condição ; modificador) {
    //código que será executado em cada repetição
}
```

ARRAYS

Os **arrays** nos permite gerar uma **coleção** de **dados ordenados**.

POSIÇÕES DENTRO DE UM ARRAY

Cada dado de um array ocupa uma posição numerada conhecida como um **índice**. A primeira posição de um array **é sempre 0**.

```
let meuArray = ['Star Wars', 'Kill Bill', 'Alien'];
                      ↑          ↑         ↑
                      0          1         2
```

Para acessar um elemento específico de um array, nomeamos o array e, **dentro** dos **colchetes**, escrevemos o **índice** que queremos acessar.

```
meuArray[2];
// acessamos o filme Alien, o índice 2 da matriz
```

ESTRUTURA DE UM ARRAY

Utilizamos colchetes `[]` para indicar o **início** e o **fim** de um array. Utilizamos vírgula `,` para **separar** seus elementos.

Dentro, podemos armazenar a quantidade de elementos que quisermos sem nos importar com o tipo de dado de cada um.

Isso quer dizer que, em um mesmo array podemos ter dados de tipo string, number, boolean, e todos os demais.

```
let meuArray = ['Star Wars', true, 23];
```