



INFINITY SCHOOL
VISUAL ART CREATIVE CENTER

AULA 06 – FUNÇÕES

O QUE IREMOS APRENDER

01 RESUMO DA AULA

02 PASSADA FUNÇÕES

03 FUNÇÕES ANÔNIMAS

04 FUNÇÕES EMBUTIDAS

05 MÃOS NO CÓDIGO



RESUMO DA AULA PASSADA

Um **array** é uma coleção ordenada de elementos, cada um identificado por um índice. Pode armazenar elementos de diferentes tipos (números, strings, objetos, etc.).

Índices: Os elementos são acessados por meio de índices começando do zero.

Inserção/Remoção: Elementos podem ser adicionados ou removidos.

Iteração: É possível percorrer todos os elementos usando loops.

FUNÇÕES

Funções em JavaScript são blocos de código que **realiza uma tarefa**, uma operação, sendo executada quando é chamada por alguém ou invocada.

Elas permitem que você agrupe um conjunto de instruções em uma única unidade lógica, tornando seu código mais organizado, modular e fácil de manter.

FUNÇÕES

EXEMPLO:



```
1  function funcaoExemplo(parametro1,parametro2){  
2      //codigo  
3  }
```

FUNÇÕES

Parâmetros (ou argumentos) são variáveis definidas dentro dos parênteses na declaração de uma função que recebem valores quando a função é invocada.



```
1 function soma(num1,num2,num3 = 0){  
2     return num1 + num2 + num3  
3 }  
4 soma(1,12) //13, pois num3 não foi passado e assumiu o valor 0  
5 soma(1,12,45) //58, num3 assumiu o valor 45  
6 soma(1) //ERRO - pela ordem, num2 não foi passado e ele é um parâmetro obrigatório
```

FUNÇÕES

Ao declarar uma função, você pode especificar um ou mais parâmetros entre parênteses. Cada parâmetro é um identificador (um nome) que representa um valor que será passado para a função quando ela for chamada.

OBS: OS PARÂMETROS NÃO SÃO OBRIGATÓRIOS

FUNÇÕES

Toda função tem acesso à palavra reservada `return`, é usada dentro de uma função para especificar o valor que a função deve retornar quando for invocada.

O retorno é o valor que será obtido quando a função for chamada, e esse valor pode ser usado em outras partes do código ou atribuído a uma variável.



```
1 // função somar que possui 2 parâmetros x e y
2 function somar(x,y){
3     // a função irá retornar a operação entre x e y
4     return x+y
5 }
6 //atribuindo valor para minha função somar
7 console.log(somar(3,5))
8 // irá mostrar 8 no console
```


FUNÇÕES

EXEMPLO:



```
1  /* No exemplo abaixo temos uma função que se chama calcularAreaTerreno,  
2  e como parâmetros temos largura e comprimento.*/  
3  
4  function calcularAreaTerreno(largura,comprimento){  
5      //Criamos uma variável chamada area que calcula a área que é a largura*comprimento  
6  
7      let  area = largura*comprimento  
8  
9      /*Utilizamos o return para finalizar a execução de uma função e especifica  
10     os valores que devem ser retonados para onde a função foi chamada.*/  
11  
12     return area  
13 }
```

FUNÇÕES

EXEMPLO:



```
1 function multiplicar(num,num2){  
2     return num*num2  
3 }  
4 document.getElementById("demo").innerHTML = multiplicar(3,2)
```

ATIVIDADE PRÁTICA

Atividade 01

Escreva uma função que permita contar o número de vogais contidas em uma string fornecida pelo usuário. Por exemplo, o usuário informa a string "Beterraba", e a função retorna o número 4 (há 4 vogais nessa palavra).

Atividade 02

Implemente uma função que receba um número como parâmetro e informe o quadrado desse número.

ATIVIDADE PRÁTICA

Atividade 03

Escreva uma função que encontre a área e o perímetro de um círculo, de acordo com o raio fornecido

Atividade 04

Escreva uma função que verifica se um número fornecido pelo usuário em um prompt é primo ou não.

FUNÇÕES ANÔNIMAS

Em JavaScript, funções anônimas são funções que não possuem um nome associado a elas. Elas são também conhecidas como funções "lambda" ou "funções sem nome". Em vez de serem declaradas com um nome identificador, as funções anônimas são definidas como expressões e podem ser atribuídas a variáveis, passadas como argumentos para outras funções ou mesmo retornadas por outras funções.

FUNÇÕES ANÔNIMAS

EXEMPLO:



```
1 //Criamos uma variável chamada soma que recebe uma função com os parâmetros a e b
2 const soma = function(a,b){
3     return a+b
4     //pedimos para retornar a soma entre a e b
5 }
6 console.log(soma(2,3)) // Saída 5
```



```
1 //Criamos uma variável chamada saudacao que recebe uma função com o parâmetro nome
2 const saudacao = function(nome){
3     return `Olá ${nome}`
4 }
5 console.log(saudacao("Raama")) // Olá Raama
```


ATIVIDADE PRÁTICA

Atividade 05

Crie uma função anônima que simule uma calculadora básica. Ela deve aceitar dois números e uma operação (adição, subtração, multiplicação, divisão) como parâmetros e retornar o resultado.

Atividade 06

Crie uma função anônima que aceite dois números como parâmetros e retorne a soma deles.

FUNÇÕES ANÔNIMAS

Em JavaScript, há várias funções nativas (também conhecidas como funções embutidas ou prontas) que estão disponíveis para uso sem a necessidade de defini-las. Aqui estão algumas das funções prontas mais comuns em JavaScript:

`console.log()`: Usada para imprimir mensagens no console.

`alert()`: Exibe uma caixa de diálogo com uma mensagem.

`prompt()`: Exibe uma caixa de diálogo que solicita a entrada do usuário.

ATIVIDADE PRÁTICA

Atividade 07

Faça um programa que use a função `valorPagamento` para determinar o valor a ser pago por uma prestação de uma conta. O programa deverá solicitar ao usuário o valor da prestação e o número de dias em atraso e passar estes valores para a função `valorPagamento`, que calculará o valor a ser pago e devolverá este valor ao programa que a chamou. O programa deverá então exibir o valor a ser pago na tela.

ATIVIDADE PRÁTICA

Atividade 07

Após a execução o programa deverá voltar a pedir outro valor de prestação e assim continuar até que seja informado um valor igual a zero para a prestação. Neste momento o programa deverá ser encerrado, exibindo o relatório do dia, que conterá a quantidade e o valor total de prestações pagas no dia. O cálculo do valor a ser pago é feito da seguinte forma. Para pagamentos sem atraso, cobrar o valor da prestação. Quando houver atraso, cobrar 3% de multa, mais 0,1% de juros por dia de atraso.

DESAFIO PRÁTICO

Faça um programa que implemente um jogo de Craps. O jogador lança um par de dados, obtendo um valor entre 2 e 12. Se, na primeira jogada, você tirar 7 ou 11, você um "natural" e ganhou. Se você tirar 2, 3 ou 12 na primeira jogada, isto é chamado de "craps" e você perdeu. Se, na primeira jogada, você fez um 4, 5, 6, 8, 9 ou 10, este é seu "Ponto". Seu objetivo agora é continuar jogando os dados até tirar este número novamente. Você perde, no entanto, se tirar um 7 antes de tirar este Ponto novamente.

SE LIGA NO CONTEÚDO DA PRÓXIMA AULA!

AULA 07 DE
JAVASCRIPT. FUNÇÕES
II

IN

INFINITY SCHOOL
VISUAL ART CREATIVE CENTER

ARROW FUNCTIONS

Arrow functions, são um tipo de sintaxe utilizada para escrever funções de forma mais condensada. A sintaxe de uma arrow function retira a palavra function e adiciona um “=>” (como uma flecha) entre o parênteses e o escopo da função. Ela funciona como uma função anônima, precisando ser associada a alguma outra estrutura de dados.



ARROW FUNCTIONS

Exemplos da ARROW FUNCTIONS em JavaScript:



```
1 // Função tradicional
2 function soma(a, b) {
3     return a + b;
4 }
5 // Arrow function equivalente
6 const somaArrow = (a, b) => a + b;
7 console.log(soma(2, 3)); // Saída: 5
8 console.log(somaArrow(2, 3)); // Saída: 5
```



INFINITY SCHOOL
VISUAL ART CREATIVE CENTER

AULA 06 – FUNÇÕES