Aula 07

Funções

Lógica de Programação com Javascript

Prof. Augusto César Oliveira
augustocesarfmo@gmail.com

SOFTEX
PERNAMBUCO

Na aula passada…

- Reforçar o conceito de estrutura de repetição;
- Conhecer o comando "while" e aprender a utilizá-lo;
- Diferenciar os comandos "for" e "while"; e
- Conhecer o comando "do… while" e aprender a utilizá-lo.

O objetivo da aula de hoje…

- Compreender o conceito de funções;
- Construir funções com parâmetros de entrada; e
- Construir funções com parâmetros de "entrada" e "saída".

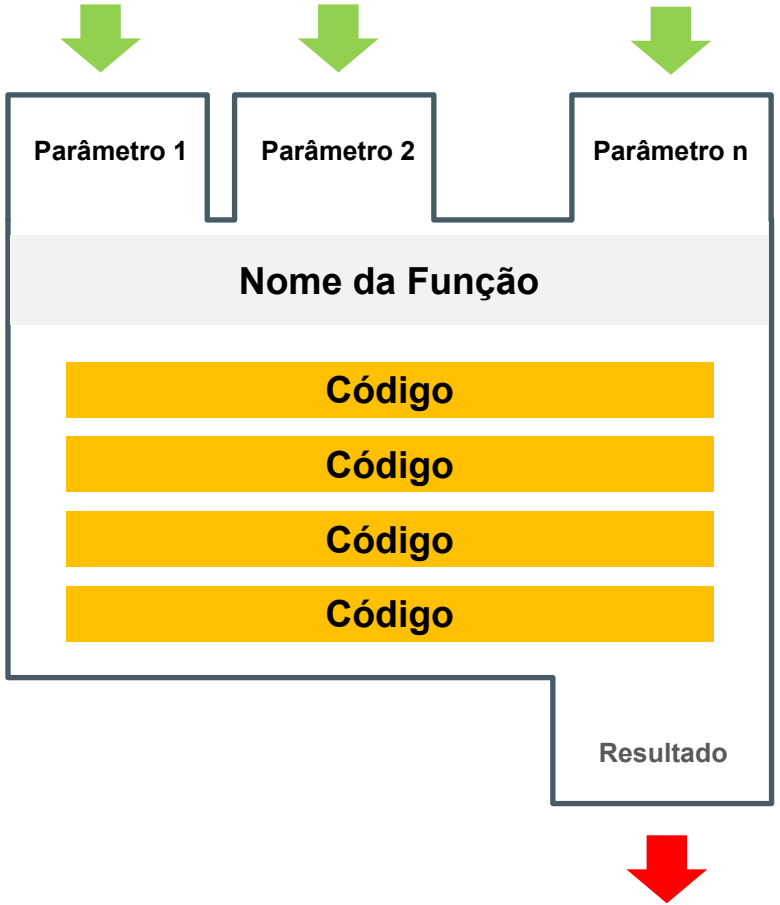
1. Funções

Funções

O que é uma função?

- Uma função é utilizada para **agrupar blocos de códigos** e **evitar a repetição deles**.

- Uma função **recebe parâmetros de entrada** e **emite uma saída**.



Exemplo de repetição de código

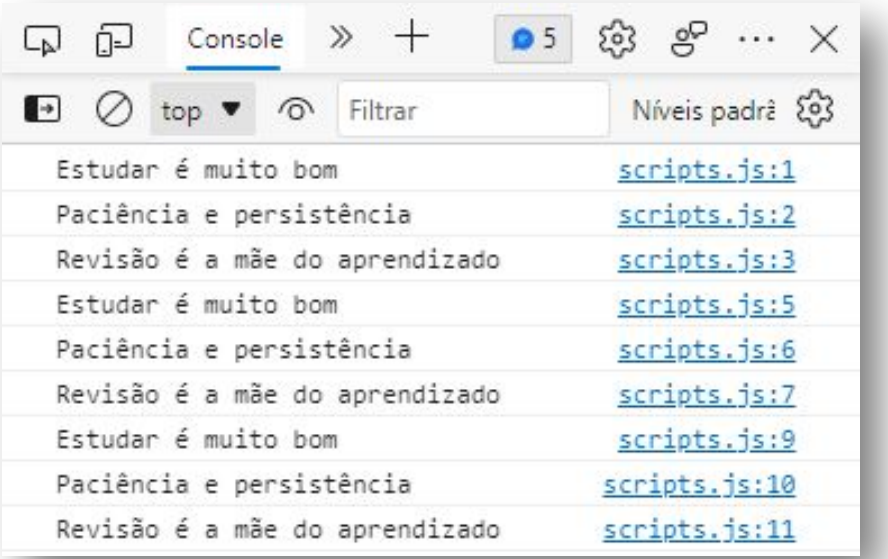
```
1   console.log('Estudar é muito bom')
2   console.log('Paciência e persistência')
3   console.log('Revisão é a mãe do aprendizado')
4
5   console.log('Estudar é muito bom')
6   console.log('Paciência e persistência')
7   console.log('Revisão é a mãe do aprendizado')
8
9   console.log('Estudar é muito bom')
10  console.log('Paciência e persistência')
11  console.log('Revisão é a mãe do aprendizado')
```

Exemplo de repetição de código

```
1  console.log('Estudar é muito bom')
2  console.log('Paciência e persistência')
3  console.log('Revisão é a mãe do aprendizado')
4
5  console.log('Estudar é muito bom')
6  console.log('Paciência e persistência')
7  console.log('Revisão é a mãe do aprendizado')
8
9  console.log('Estudar é muito bom')
10 console.log('Paciência e persistência')
11 console.log('Revisão é a mãe do aprendizado')
```

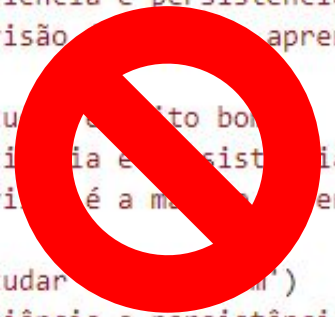
Exemplo de repetição de código

```
1  console.log('Estudar é muito bom')
2  console.log('Paciência e persistência')
3  console.log('Revisão é a mãe do aprendizado')
4
5  console.log('Estudar é muito bom')
6  console.log('Paciência e persistência')
7  console.log('Revisão é a mãe do aprendizado')
8
9  console.log('Estudar é muito bom')
10 console.log('Paciência e persistência')
11 console.log('Revisão é a mãe do aprendizado')
```

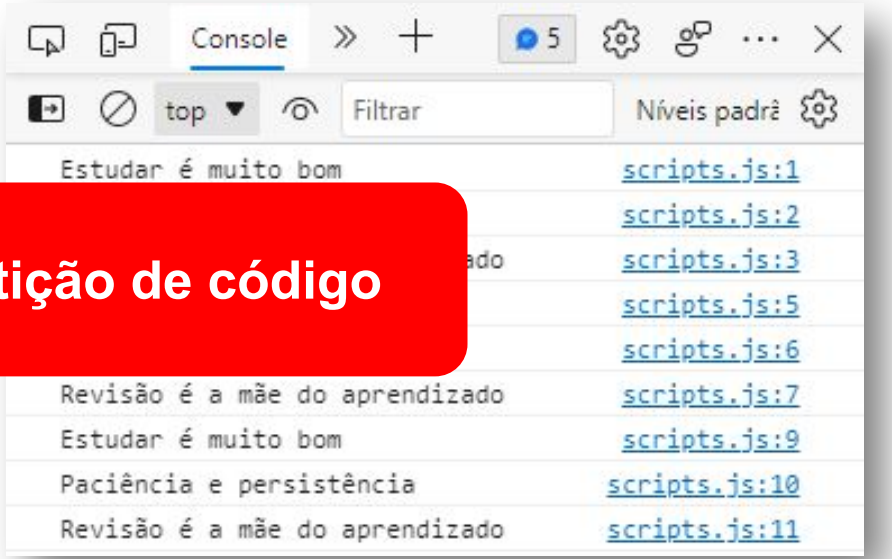


Exemplo de repetição de código

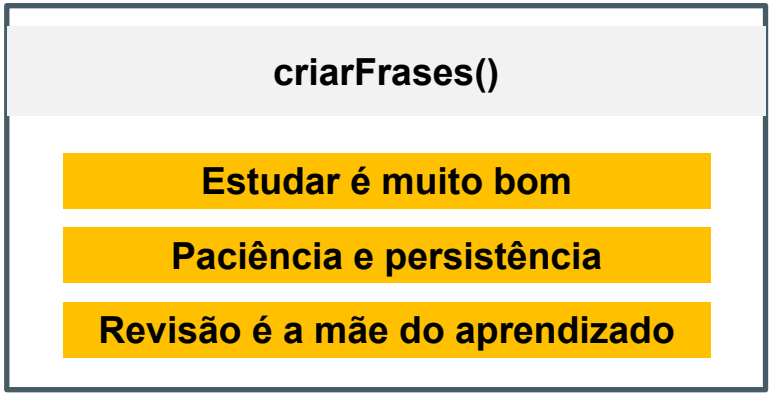
```
1   console.log('Estudar é muito bom')
2   console.log('Paciência e persistência')
3   console.log('Revisão                aprendizado')
4
5   console.log('Estu          ito bo
6   console.log('Paci          ia e          ist          ia')
7   console.log('Revi          é a m                  endiz
8
9   console.log('Estudar                          m')
10  console.log('Paciência e persistência')
11  console.log('Revisão é a mãe do aprendizado')
```



Repetição de código



Uma função "sem parâmetros" e sem "saída"



Uma função "sem parâmetros" e sem "saída"

criarFrases()

Estudar é muito bom

Paciência e persistência

Revisão é a mãe do aprendizado

```
1  function criarFrases(){
2      console.log('Estudar é muito bom')
3      console.log('Paciência e persistência')
4      console.log('Revisão é a mãe do aprendizado')
5  }
```

Declaração de uma função

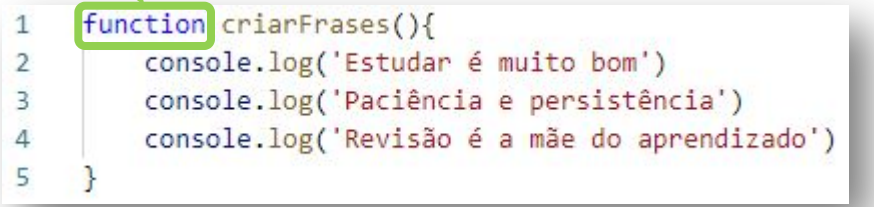
criarFrases()

Estudar é muito bom

Paciência e persistência

Revisão é a mãe do aprendizado

Iniciando a função



```
1  function criarFrases(){
2      console.log('Estudar é muito bom')
3      console.log('Paciência e persistência')
4      console.log('Revisão é a mãe do aprendizado')
5  }
```

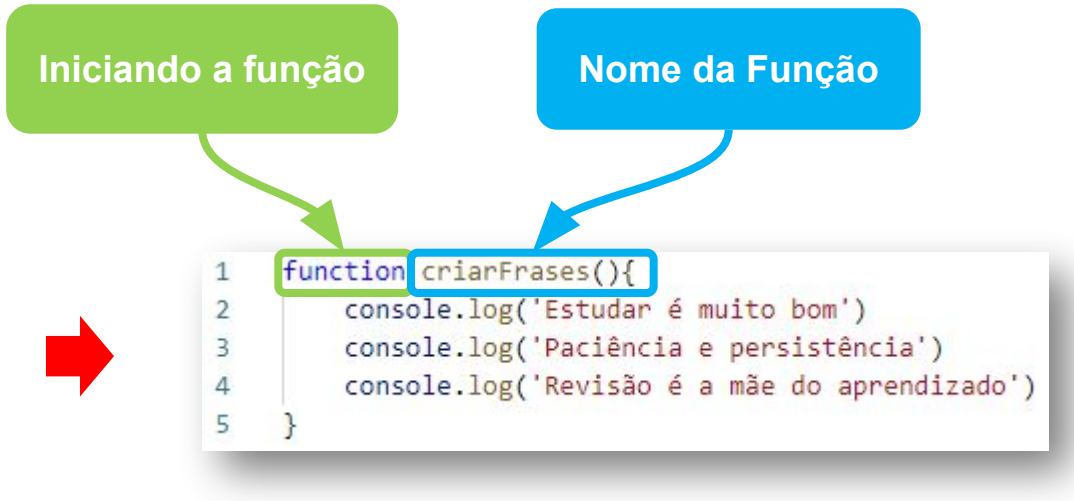
O nome da função

criarFrases()

Estudar é muito bom

Paciência e persistência

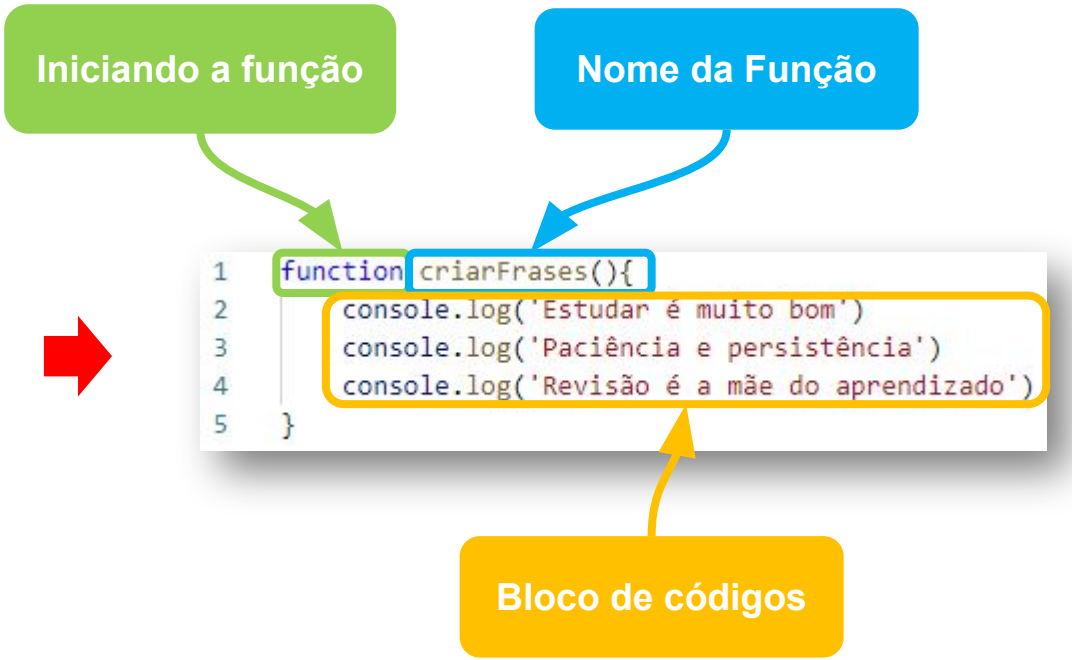
Revisão é a mãe do aprendizado



O "corpo" da função

criarFrases()

- **Estudar é muito bom**
- **Paciência e persistência**
- **Revisão é a mãe do aprendizado**



Uma função para evitar a "repetição" de código

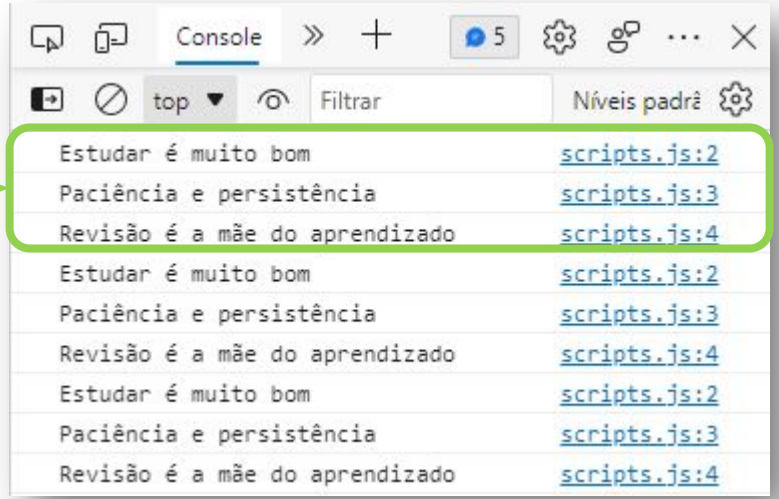
```
1  function criarFrases(){
2      console.log('Estudar é muito bom')
3      console.log('Paciência e persistência')
4      console.log('Revisão é a mãe do aprendizado')
5  }
6
7  criarFrases()
8  criarFrases()
9  criarFrases()
```

Uma função para evitar a "repetição" de código

```
1  function criarFrases(){
2      console.log('Estudar é muito bom')
3      console.log('Paciência e persistência')
4      console.log('Revisão é a mãe do aprendizado')
5  }
6
7  criarFrases()
8  criarFrases()
9  criarFrases()
```

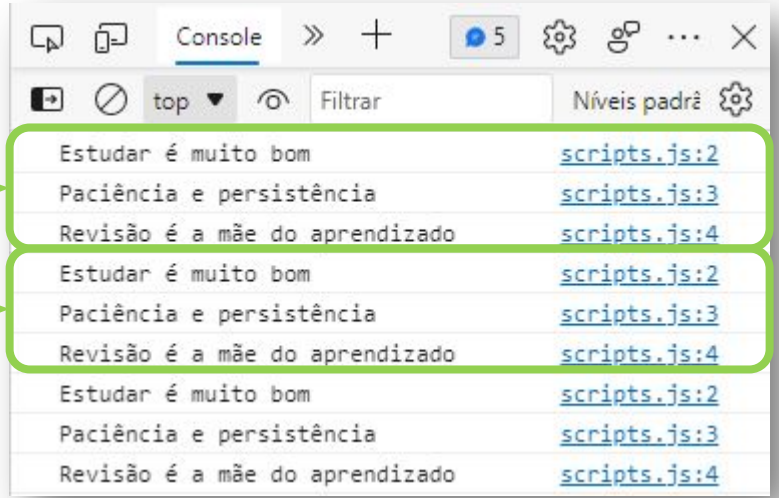
Uma função para evitar a "repetição" de código

```
1  function criarFrases(){
2      console.log('Estudar é muito bom')
3      console.log('Paciência e persistência')
4      console.log('Revisão é a mãe do aprendizado')
5  }
6
7  criarFrases()
8  criarFrases()
9  criarFrases()
```



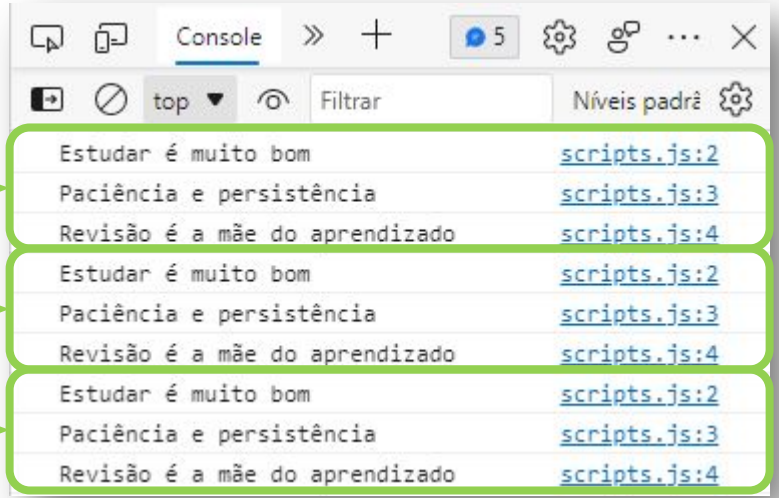
Uma função para evitar a "repetição" de código

```
1  function criarFrases(){
2      console.log('Estudar é muito bom')
3      console.log('Paciência e persistência')
4      console.log('Revisão é a mãe do aprendizado')
5  }
6
7  criarFrases()
8  criarFrases()
9  criarFrases()
```



Uma função para evitar a "repetição" de código

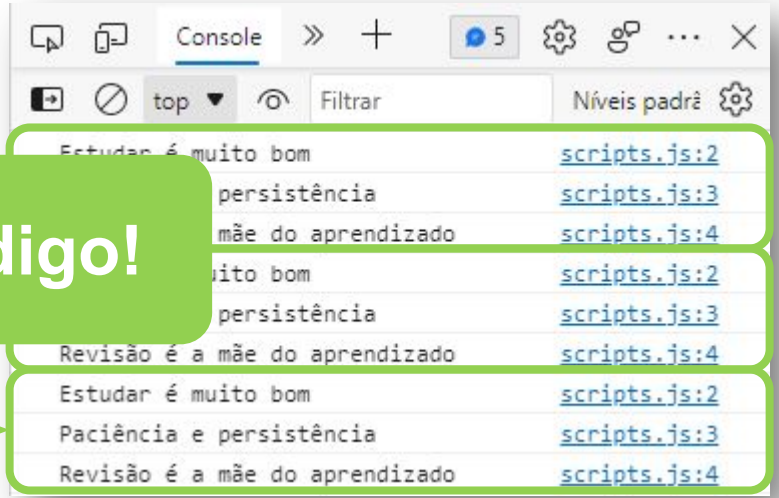
```
1  function criarFrases(){
2      console.log('Estudar é muito bom')
3      console.log('Paciência e persistência')
4      console.log('Revisão é a mãe do aprendizado')
5  }
6
7  criarFrases()
8  criarFrases()
9  criarFrases()
```



Uma função para evitar a "repetição" de código

```
1  function criarFrases(){
2      console.log('Estudar é muito bom')
3      console.log('Paciência e
4      console.log('Revisão é a
5  }
6
7  criarFrases()
8  criarFrases()
9  criarFrases()
```

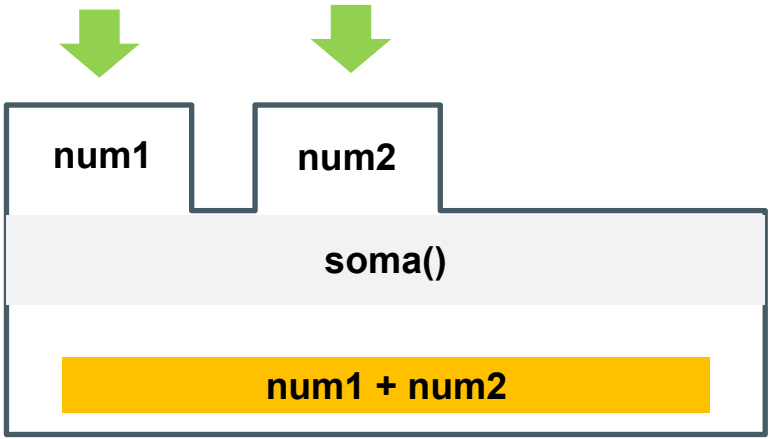
Reuso de código!



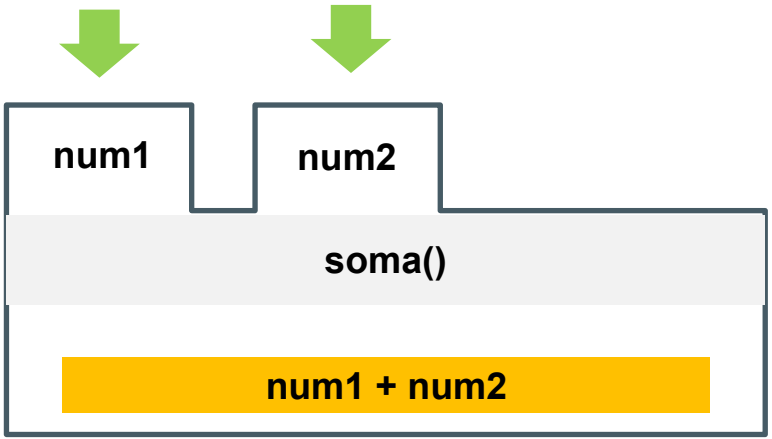
2.
Parâmetros de "função"

Funções

Uma função com parâmetros de "entrada"



Uma função com parâmetros de "entrada"



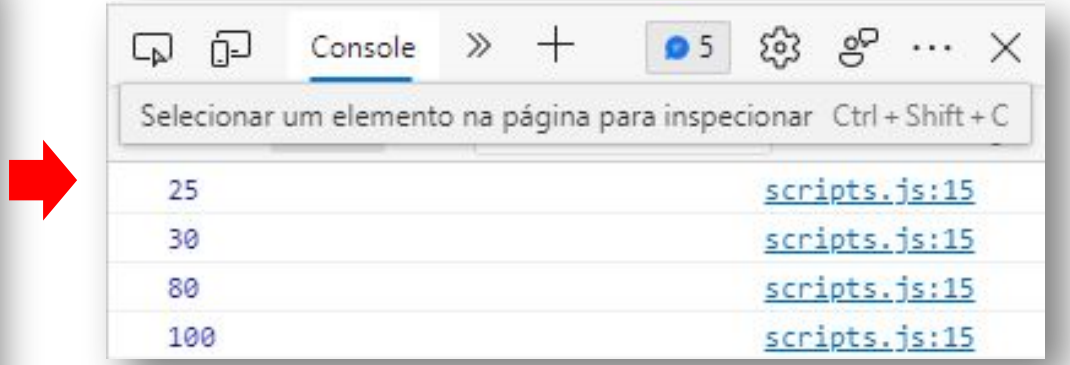

```
13  function soma(num1, num2){
14      console.log(num1 + num2)
15  }
```

A função "soma"

```
13  function soma(num1, num2){
14      console.log(num1 + num2)
15  }
16
17  soma(10, 15)
18  soma(10, 20)
19  soma(30, 50)
20  soma(50, 50)
```

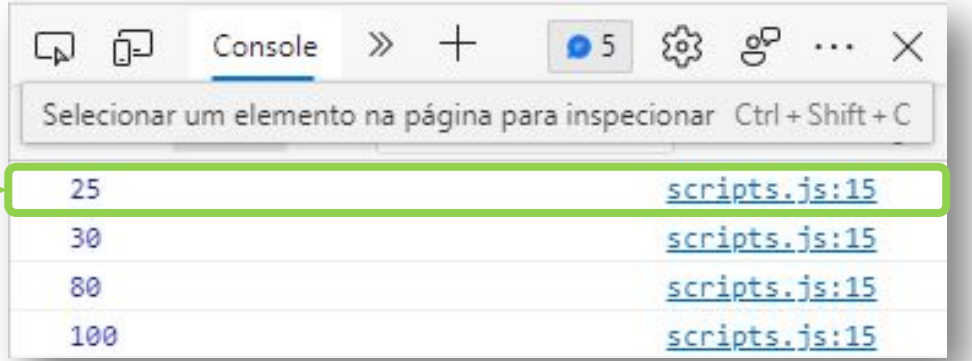
A função "soma"

```
13  function soma(num1, num2){
14      console.log(num1 + num2)
15  }
16
17  soma(10, 15)
18  soma(10, 20)
19  soma(30, 50)
20  soma(50, 50)
```



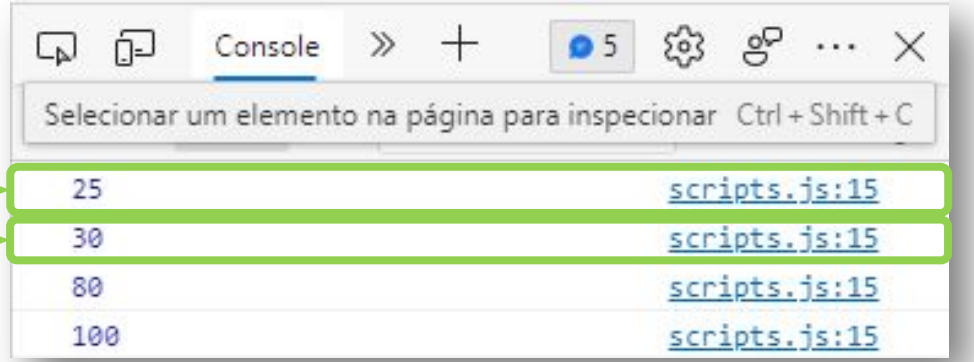
Reúso de funções

```
13  function soma(num1, num2){
14      console.log(num1 + num2)
15  }
16
17  soma(10, 15)
18  soma(10, 20)
19  soma(30, 50)
20  soma(50, 50)
```



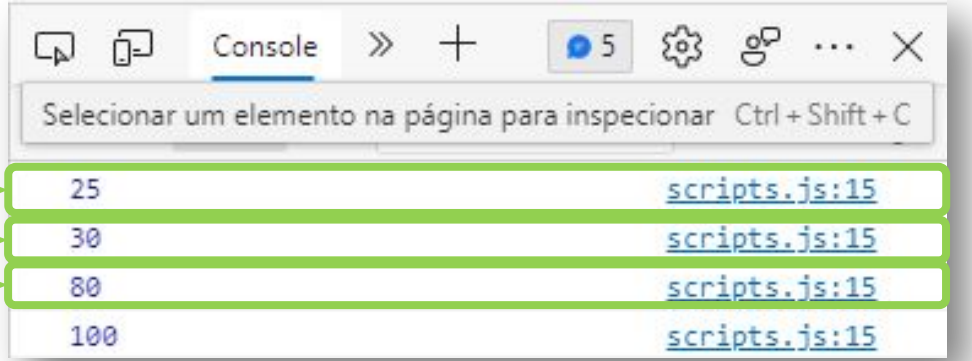
Reúso de funções

```
13  function soma(num1, num2){
14      console.log(num1 + num2)
15  }
16
17  soma(10, 15)
18  soma(10, 20)
19  soma(30, 50)
20  soma(50, 50)
```



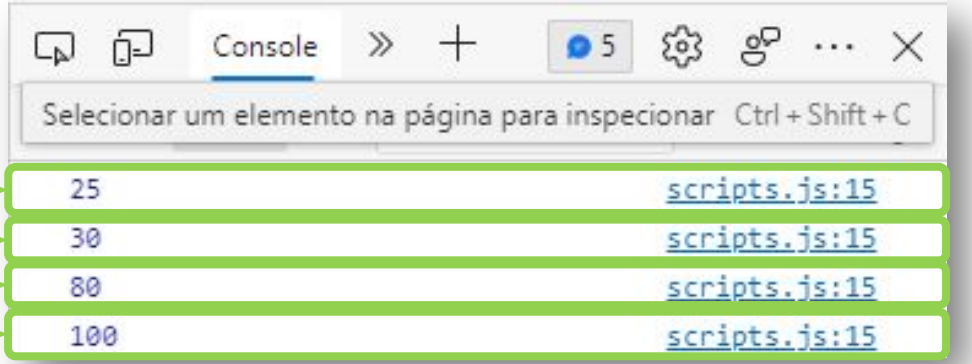
Reúso de funções

```
13  function soma(num1, num2){
14      console.log(num1 + num2)
15  }
16
17  soma(10, 15)
18  soma(10, 20)
19  soma(30, 50)
20  soma(50, 50)
```



Reúso de funções

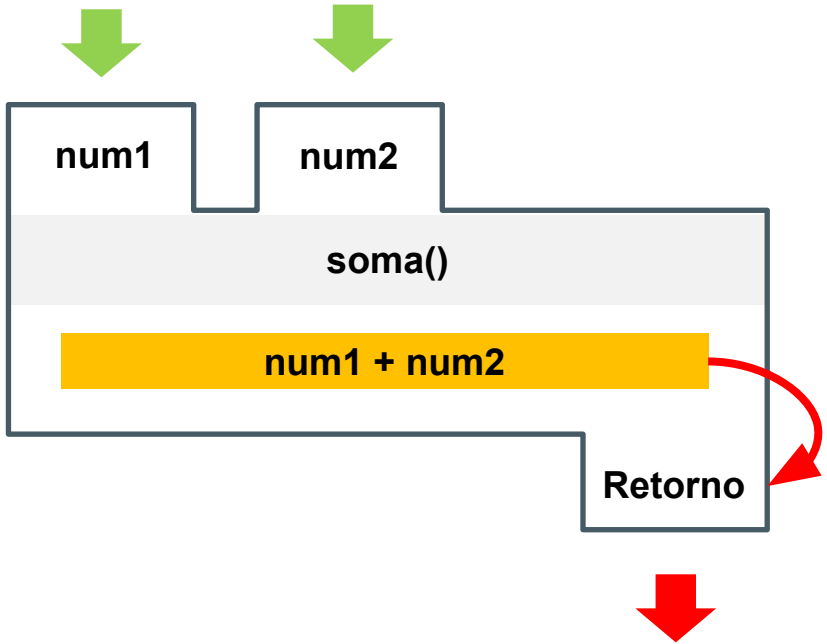
```
13  function soma(num1, num2){
14      console.log(num1 + num2)
15  }
16
17  soma(10, 15)
18  soma(10, 20)
19  soma(30, 50)
20  soma(50, 50)
```



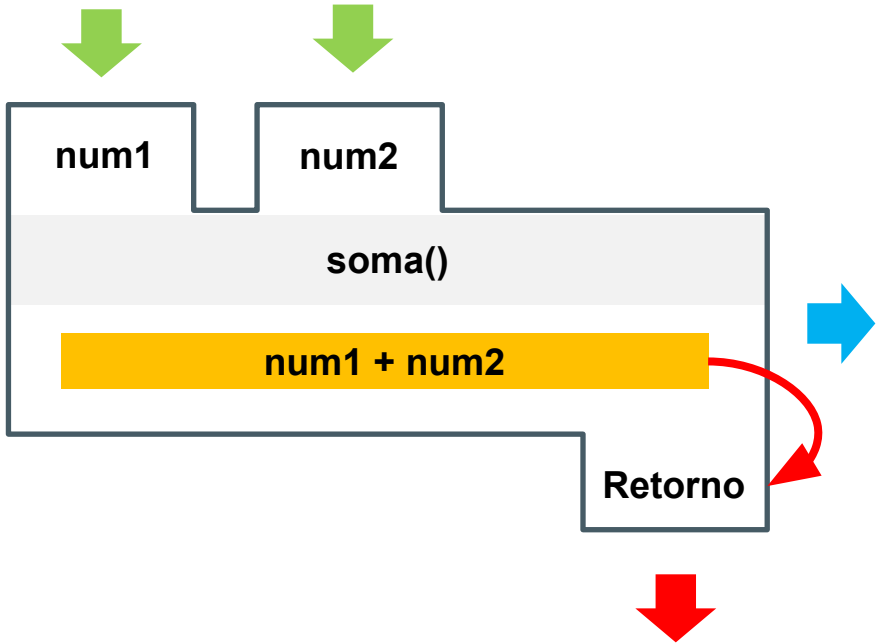
3.
O retorno da "função"

Funções

Uma função com parâmetros de "entrada" e "saída"



Uma função com parâmetros de "entrada" e "saída"



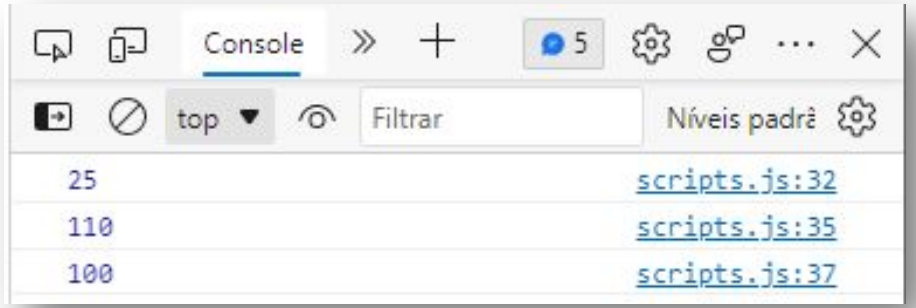

```
27    function soma(num1, num2){
28        return num1 + num2
29    }
```

Um exemplo do "retorno" da função

```
27  function soma(num1, num2){
28      return num1 + num2
29  }
30
31  teste1 = soma(10, 15)
32  console.log(teste1)
33
34  teste2 = soma(10, 20) + soma(30, 50)
35  console.log(teste2)
36
37  console.log(soma(50, 50))
```

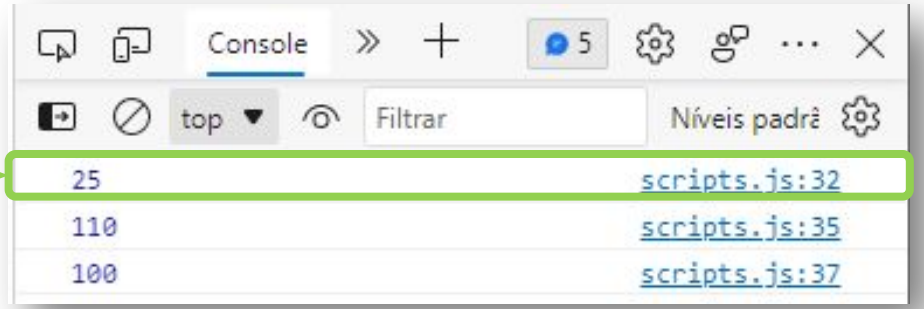
Um exemplo do "retorno" da função

```
27  function soma(num1, num2){
28      return num1 + num2
29  }
30
31  teste1 = soma(10, 15)
32  console.log(teste1)
33
34  teste2 = soma(10, 20) + soma(30, 50)
35  console.log(teste2)
36
37  console.log(soma(50, 50))
```



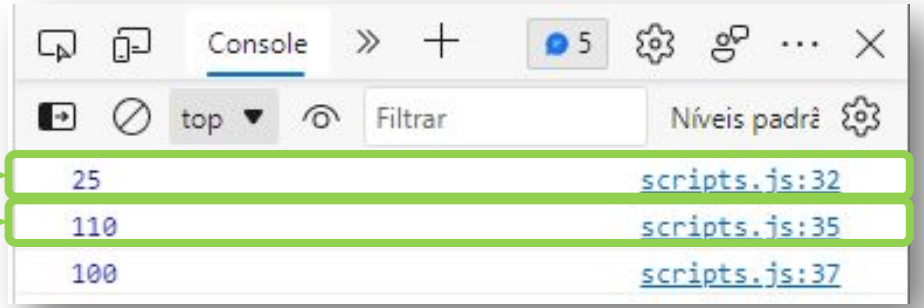
Um exemplo do "retorno" da função

```
27  function soma(num1, num2){
28      return num1 + num2
29  }
30
31  teste1 = soma(10, 15)
32  console.log(teste1)
33
34  teste2 = soma(10, 20) + soma(30, 50)
35  console.log(teste2)
36
37  console.log(soma(50, 50))
```



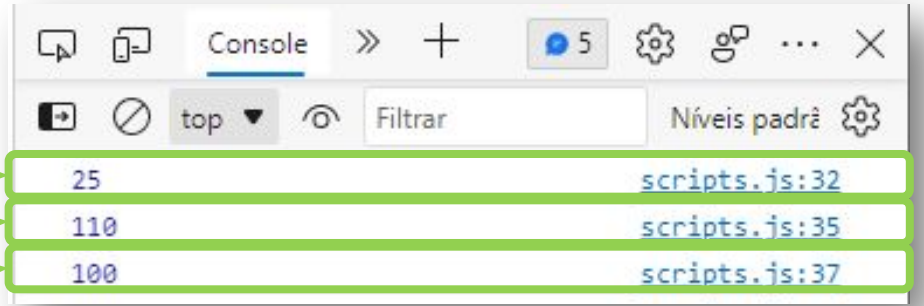
Um exemplo do "retorno" da função

```
27  function soma(num1, num2){
28      return num1 + num2
29  }
30
31  teste1 = soma(10, 15)
32  console.log(teste1)
33
34  teste2 = soma(10, 20) + soma(30, 50)
35  console.log(teste2)
36
37  console.log(soma(50, 50))
```



Um exemplo do "retorno" da função

```
27  function soma(num1, num2){
28      return num1 + num2
29  }
30
31  teste1 = soma(10, 15)
32  console.log(teste1)
33
34  teste2 = soma(10, 20) + soma(30, 50)
35  console.log(teste2)
36
37  console.log(soma(50, 50))
```



4.
Considerações finais

Funções

O que aprendemos hoje?

- O conceito de funções;
- Construir funções com parâmetros de entrada; e
- Construir funções com parâmetros de "entrada" e "saída".

Próxima aula…

- Introdução ao HTML.

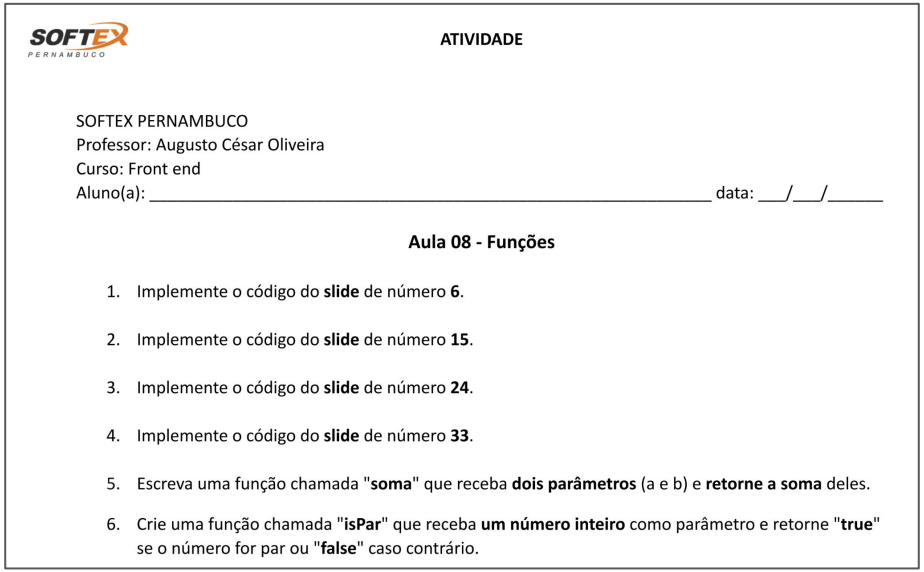
5.
Exercício de fixação

Google Classroom

Exercício de fixação

Funções

- **Link da atividade**: clique aqui.



SOFTEX PERNAMBUCO

ATIVIDADE

SOFTEX PERNAMBUCO
Professor: Augusto César Oliveira
Curso: Front end
Aluno(a): _____ data: ___/___/_____

Aula 08 - Funções

1. Implemente o código do **slide** de número **6**.
2. Implemente o código do **slide** de número **15**.
3. Implemente o código do **slide** de número **24**.
4. Implemente o código do **slide** de número **33**.
5. Escreva uma função chamada "**soma**" que receba **dois parâmetros** (a e b) e **retorne a soma** deles.
6. Crie uma função chamada "**isPar**" que receba **um número inteiro** como parâmetro e retorne "**true**" se o número for par ou "**false**" caso contrário.

Aula 07

Funções

Lógica de Programação com Javascript

Prof. Augusto César Oliveira
augustocesarfmo@gmail.com

SOFTEX
PERNAMBUCO