INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PARAÍBA

JavaScript

(Funções)

Prof. Mr. Fabio Abrantes Diniz
fabio.abrantes.diniz@gmail.com

Funções

Funções

- São **blocos de código** que realizam uma

tarefa específica e podem ser reutilizados.
  - Toda função é um objeto F**unction**
- **Vantagens**:
  - Evitam repetição de código
  - Facilitam manutenção
  - Permitem reutilização

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Fun%C3%A7%C3%B5es

Sintaxe da Função

Sintaxe da Função

```
function nomeDaFuncao(parâmetros) {
  // corpo da função
  return resultado;
}
```

- **Palavra reservada (function)**
- **Nome da função**
- **Lista de Parâmetros separados por vírgula**
- **Instruções**
- **Retorno de algum valor (return)**
  - ✓ Valor específico retornado pelo comando return
  - ✓ Ou undefined se não usar o comando return

Funções

- **Exemplo**

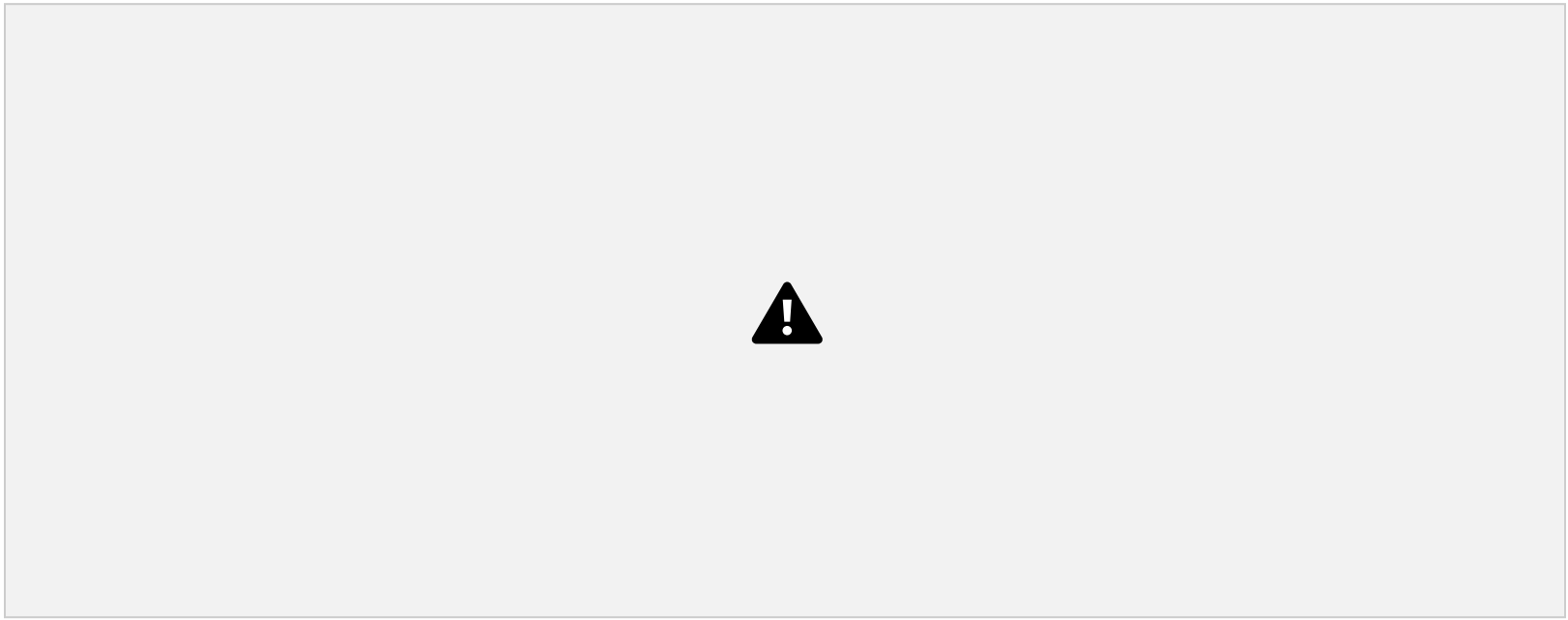
```
function saudacao(nome) {
  return "Olá, " + nome + "!";
}

console.log(saudacao("Maria")); // Olá, Maria!
```

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Fun%C3%A7%C3%B5es

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PARAÍBA

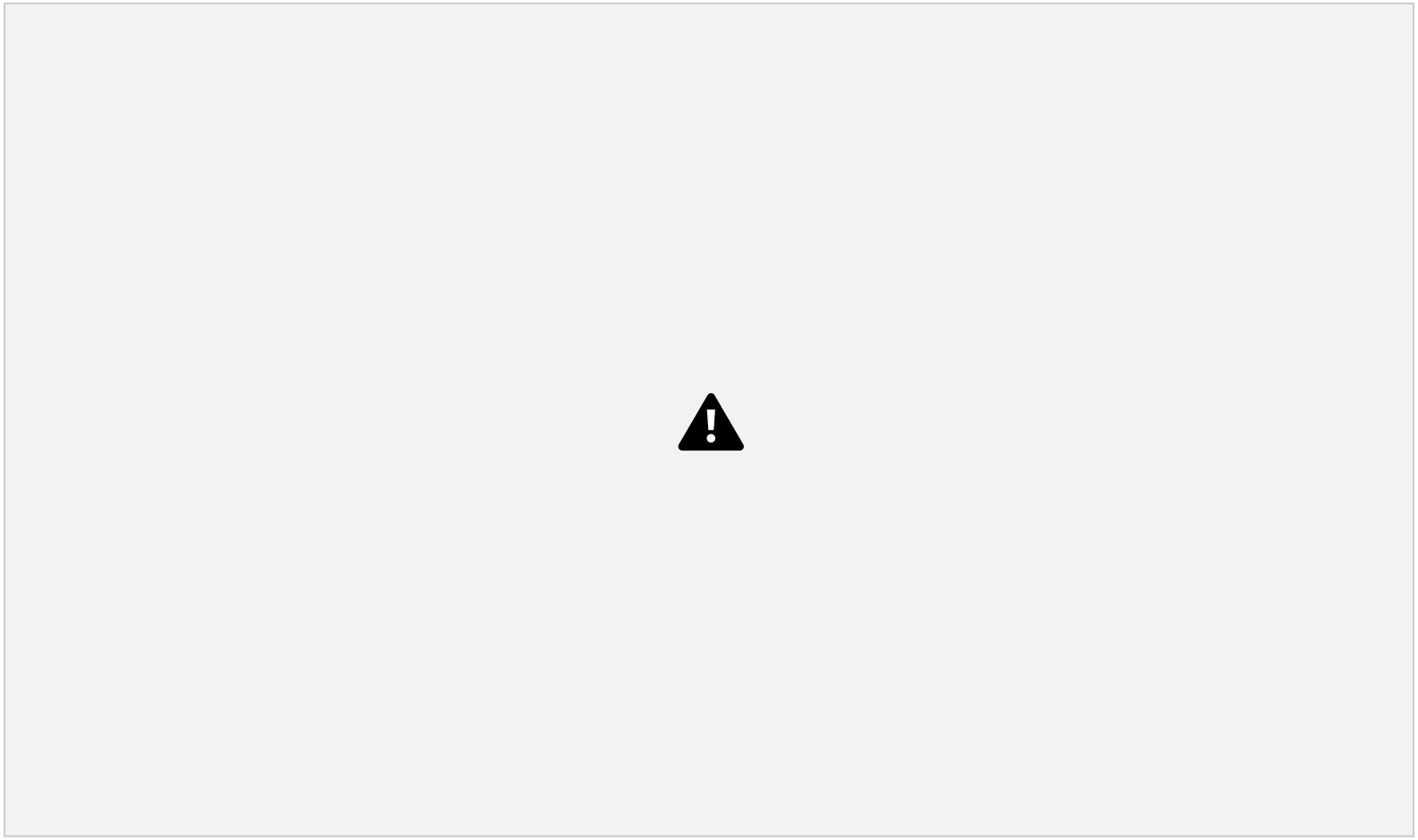
3 tipos de Funções




INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PARAÍBA

Funções

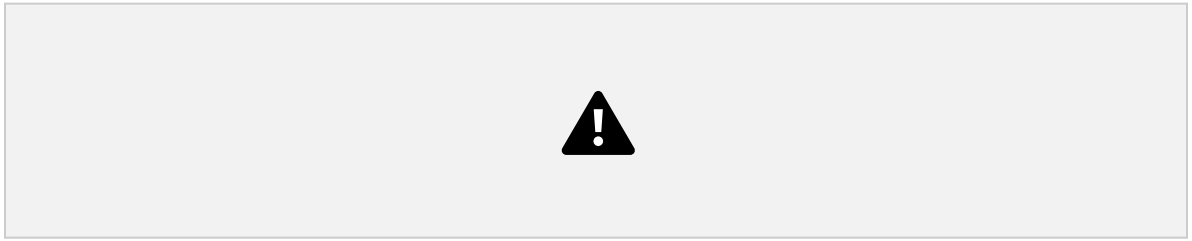
declarada



Funções Expressa

- Expressão de Função
  - Uma função anônima sendo atribuída a uma variável;

- Mas pode ser atribuída uma função nomeada

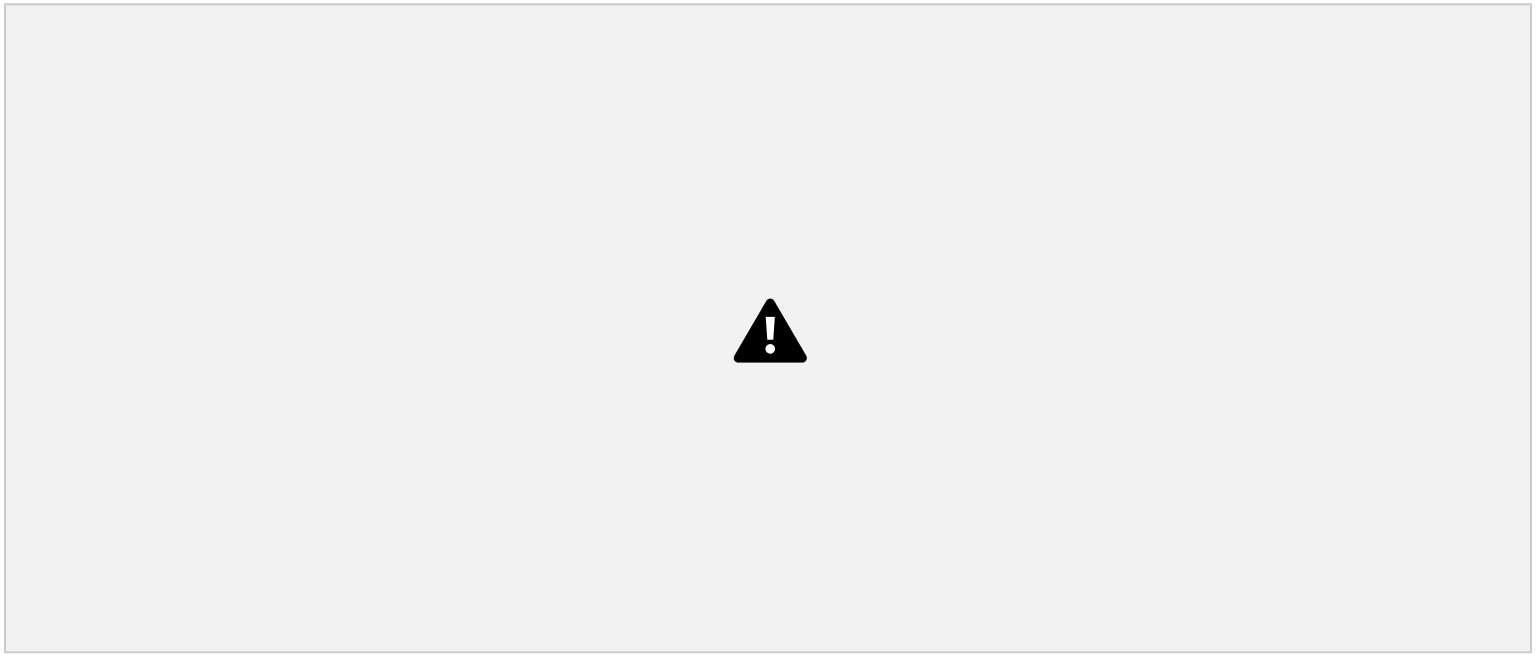


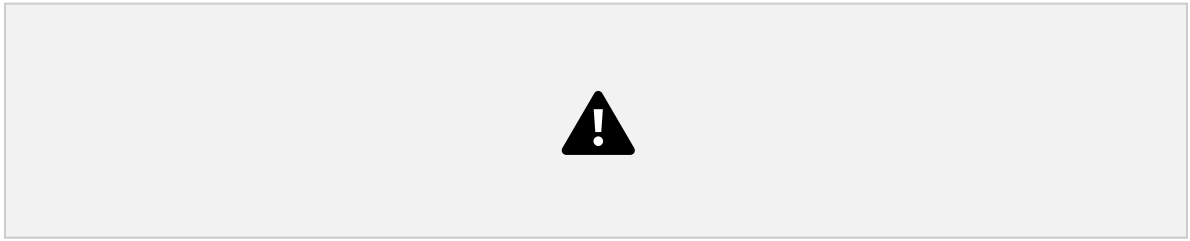
Chamando Funções

- Funções Executando sua instruções

Sintaxe

✓**nomeFuncao**(**argumentos**);

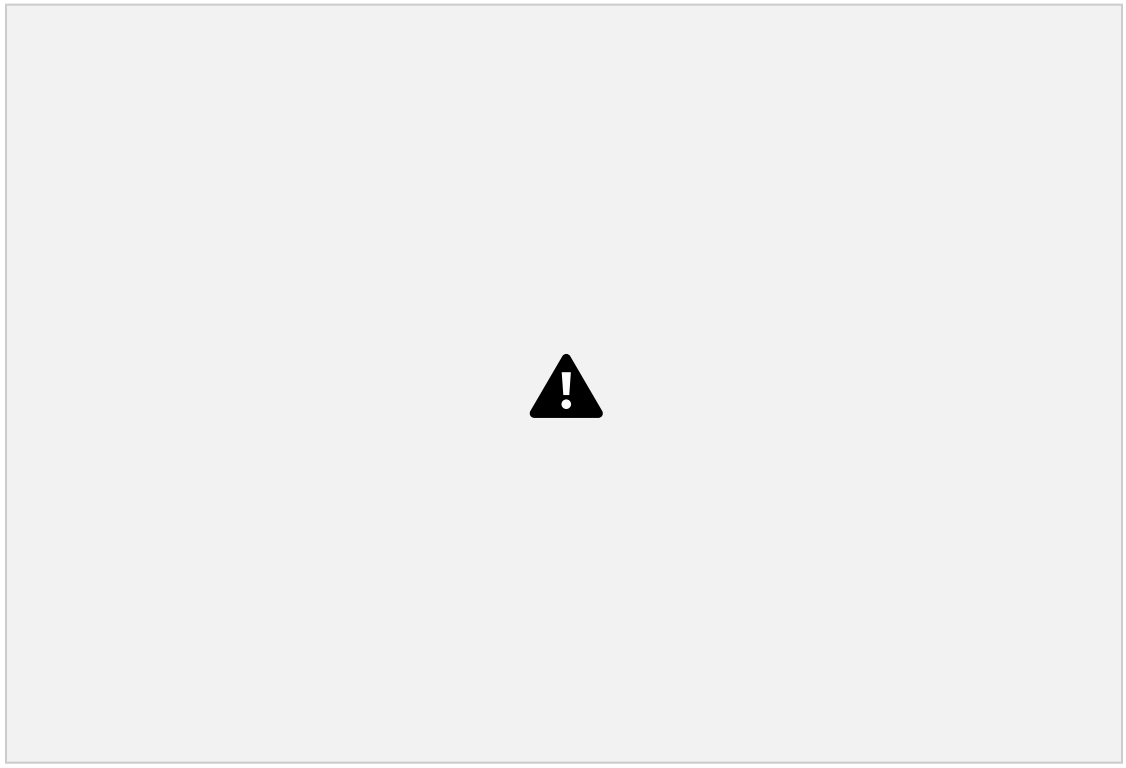




Chamando Funções

- **Sintaxe**
  - **nomeFuncao**(**argumentos**);

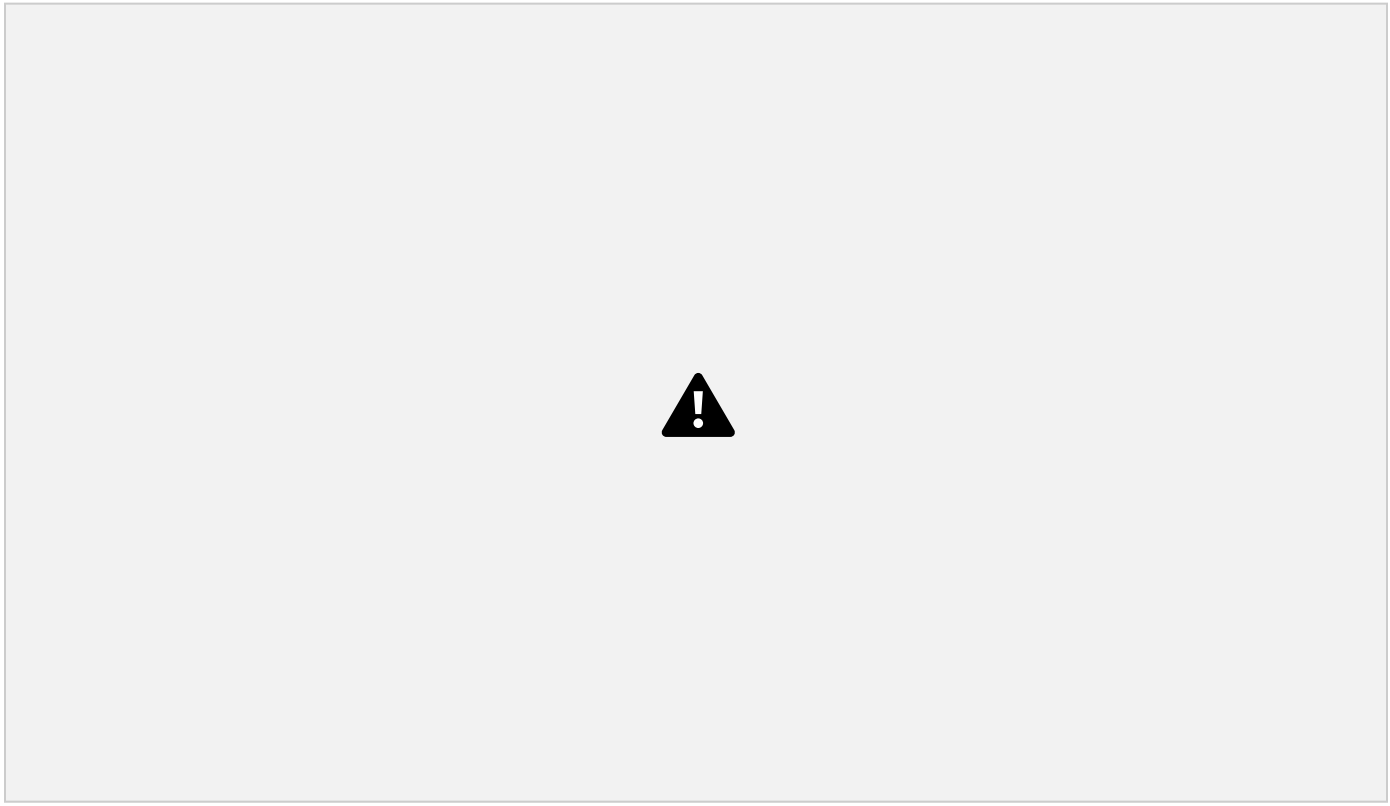
✓Os argumentos pode **objetos** e até **outras funções**



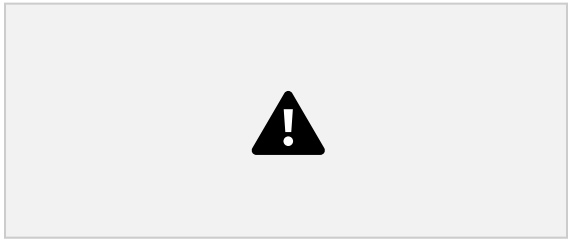
Funções Alta ordem (

callback)

- É uma função passada como **argumento** ou **retorna** outra função
  - Passamos o nome da função (referência)

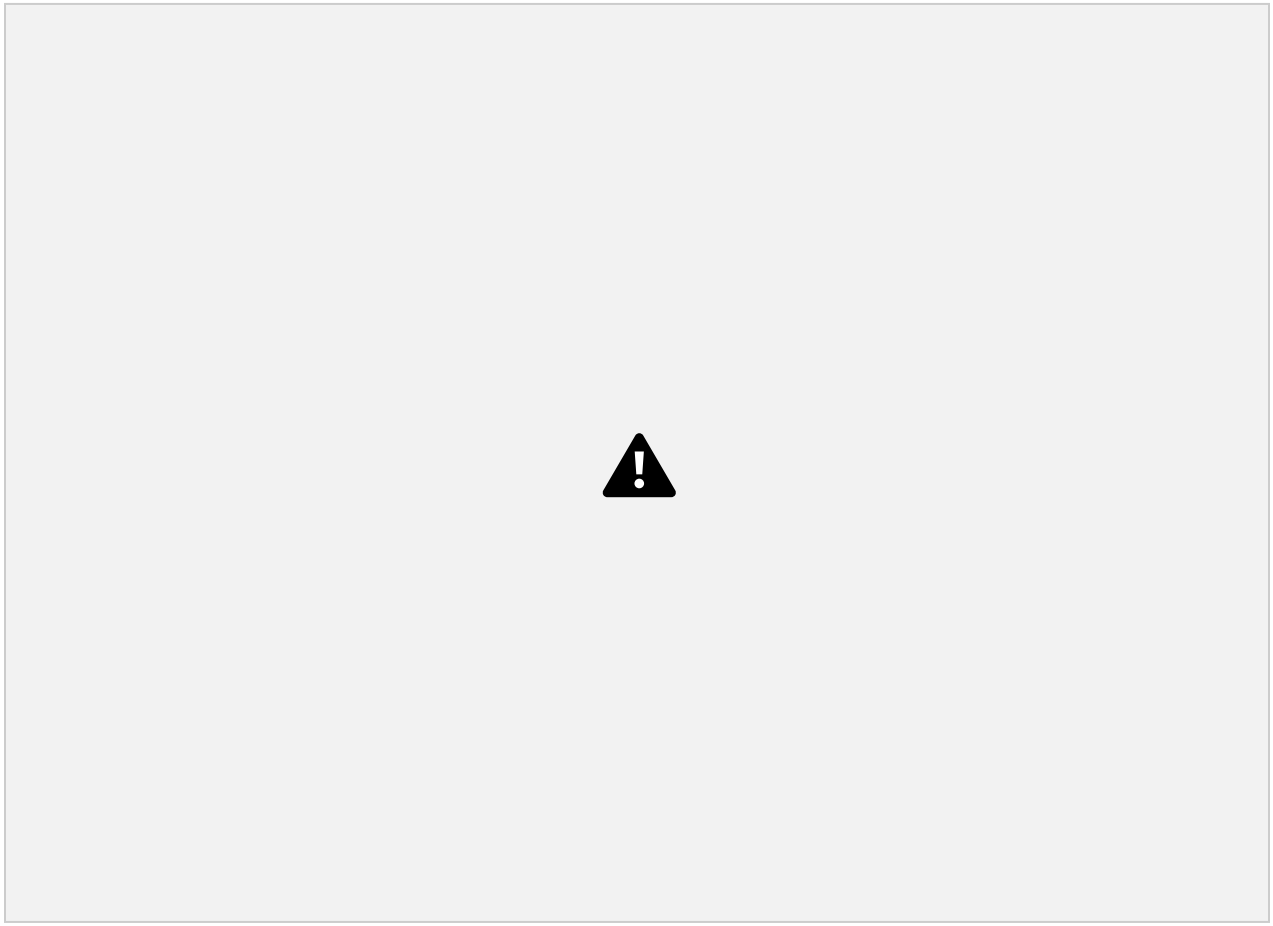


Funções Alta ordem (



callback)

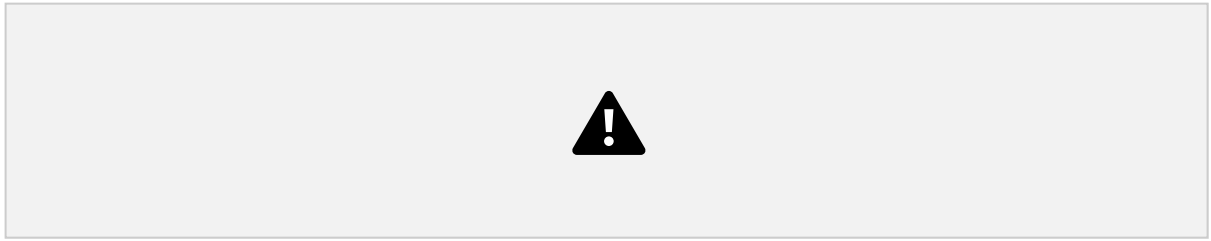
- **Exemplo que retorna outra função**



 Arrow Function (ES6)   

- Uma sintaxe mais curta
  - **Sintaxe Básica**

  - **Exemplos**

Escopo de Variáveis

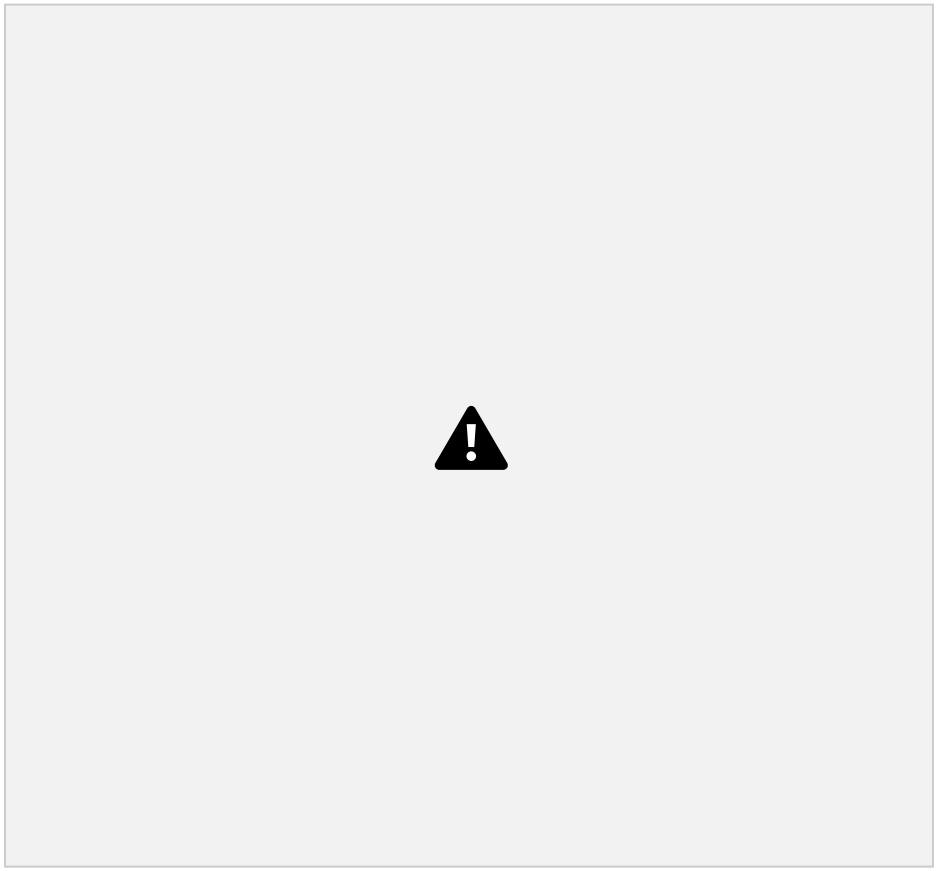
- Globais
  - Declaradas no âmbito mais amplo possível

✓Fora de funções, objetos, classes

- Locais
  - Declaradas em lugares mais dimensionados
    - ✓Escopo dentro da função, objetos, classes, blocos de for,if,while,...

Escopo de Variáveis

- **Exemplo**



Escopo de Variáveis

- var x let
  - **Let**: se for declarado dentro de bloco será acessível somente neste bloco
  - **Var** : se for declarado dentro de bloco poderá ser acessível fora do bloco

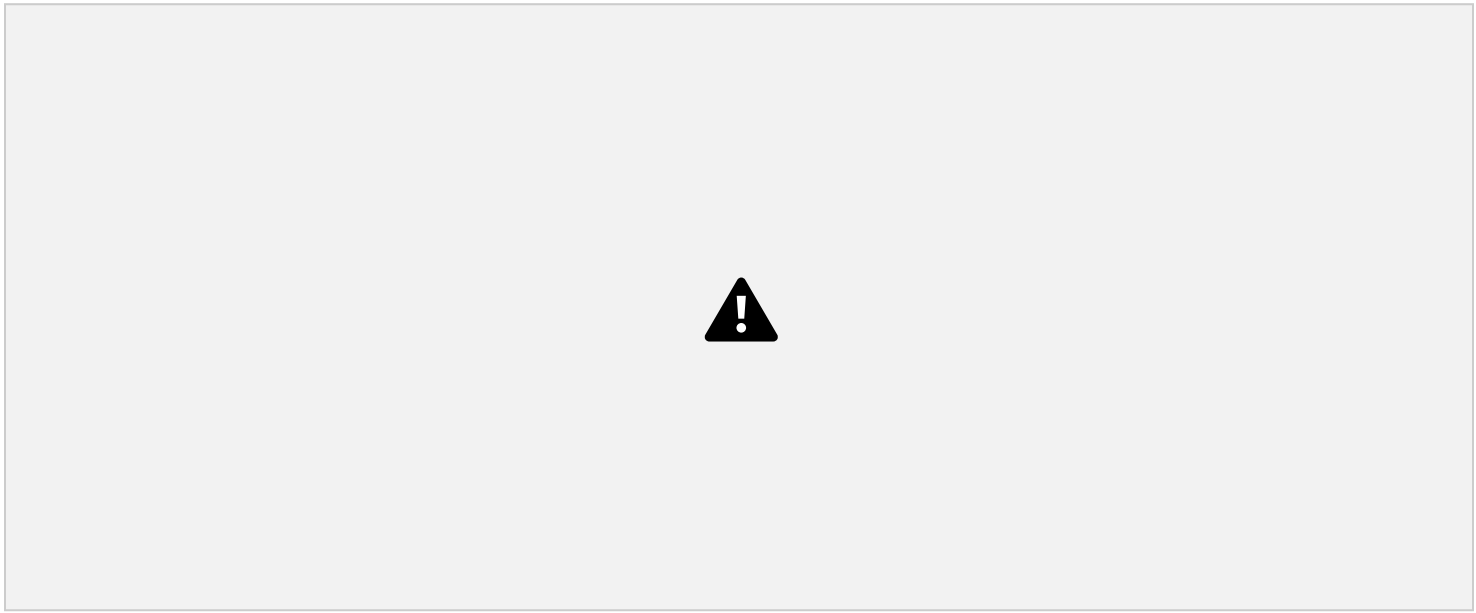
ERROR

Funções Aninhadas

- Funções definidas **dentro** de outras funções
  - função interna **só é acessível dentro da**

função externa (escopo local).

✓Uteis para **encapsular lógica** e **evitar poluir o escopo global**.



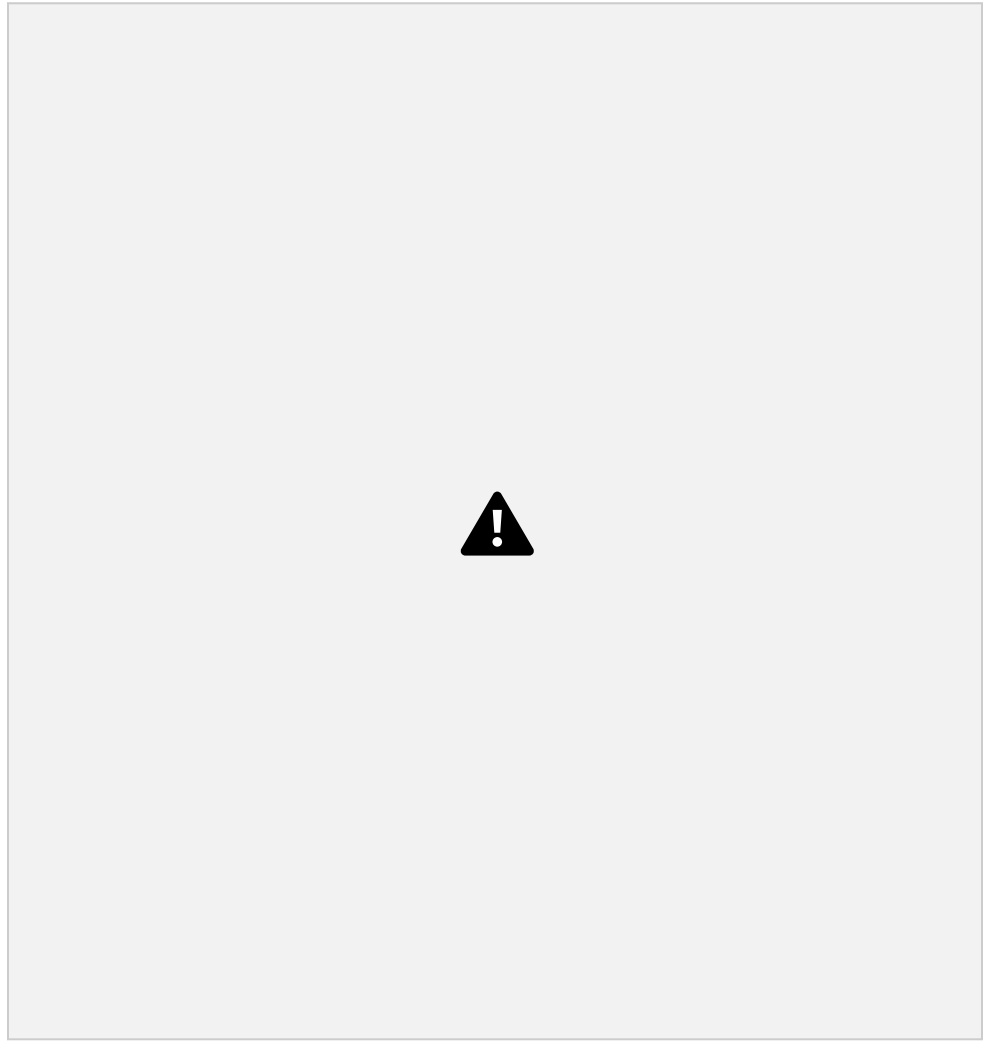
Funções Aninhadas

- **Closures e Escopo**
  - Lembra do valor de qualquer variável que esteja no escopo onde são definidas
    - ✓mesmo após a função externa ter terminado.

- Enquanto a função **externa**
  - **Não** pode usar os **argumentos** e **variáveis** da função interna.

Funções Aninhadas

- **Closures e Escopo**



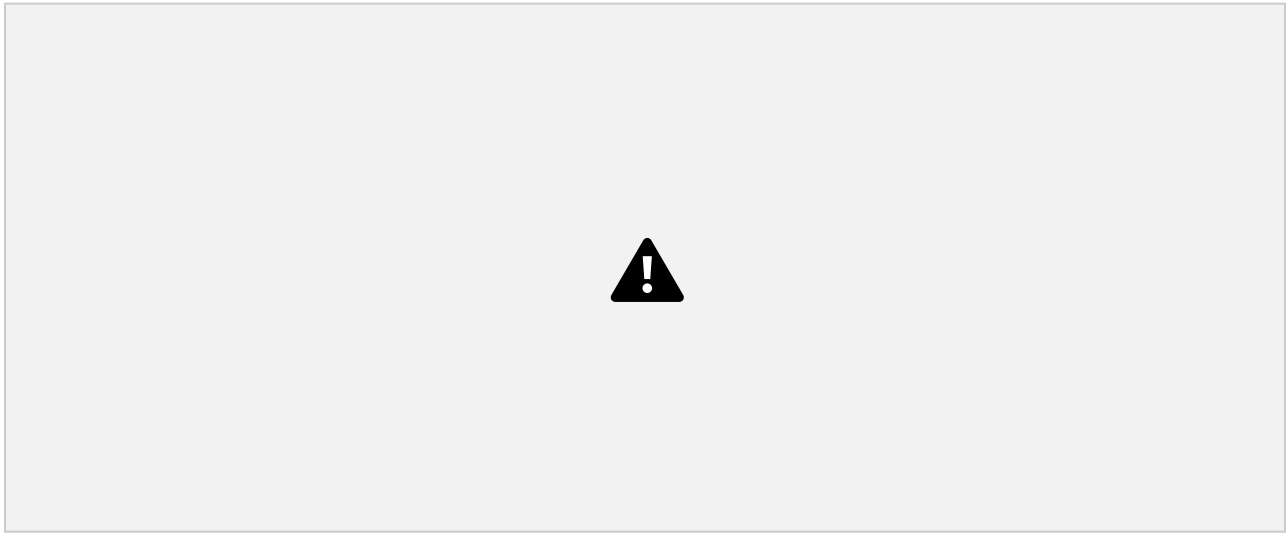
Funções Aninhadas

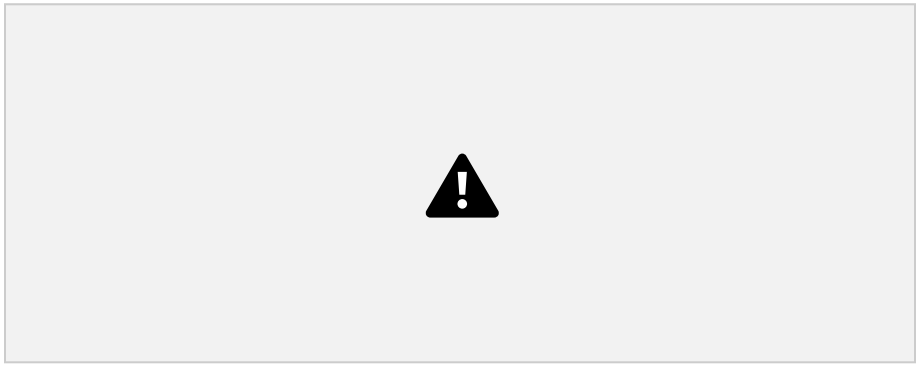
- **Quando usar?**
  - Para **organizar** melhor o código.
  - Uma função é **usada apenas dentro de outra**.
  - Para **manter variáveis privadas**
    - ✓(usando closures).
  - Em **fábricas de funções**

✓funções que retornam funções.

Funções

- Parâmetros **Opcionais** e Valores **Padrão**. ▪ Evita **undefined** e torna a função mais robusta.





Escopo de Variáveis 

Funções - Métodos



- São funções atribuídas em objetos