

Objetivo:

- I. Definição de função;
- II. Função sem parâmetros e sem retorno;
- III. Função que recebe um parâmetro e não tem retorno;
- IV. Função que recebe dois parâmetros e não tem retorno;
- V. Função que recebe dois parâmetros e tem retorno.

Observação: Para fazer os exercícios e exemplos recomenda-se o uso do VS Code ou da interface de programação online <https://replit.com/>.

I. Definição de função

Uma função é um bloco de instruções que é executado somente quando invocado e pode ser invocado inúmeras vezes, ou seja, gera modularidade e reutilização de código. Constitui boa prática organizar o código em funções. Uma função possui a seguinte notação:

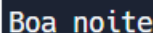
```
function nome da função( parâmetros separados por vírgulas ) {  
    instruções do bloco  
    return indica que é para sair da função e retornar um valor  
}
```

O corpo da função é delimitado pelo par de chaves. A instrução **return** é opcional, mas se ela existir é necessário ter um valor à direita da palavra reservada **return**.

II. Função sem parâmetros e sem retorno

A função **msg** não recebe parâmetros e não tem retorno. Para chamar a função temos de fornecer o nome da função seguido de um par de parênteses:

```
function msg(){  
    console.log("Boa noite")  
}  
  
//chamada da função msg  
msg( )
```

**III. Função que recebe um parâmetro e não tem retorno**

A função **saudacao** recebe um parâmetro e não tem retorno. Para chamar a função temos de colocar um valor dentro dos parênteses. Na chamada **saudacao("Ana")**, o valor **"Ana"** será colocado na variável **nome** e na chamada **saudacao("Pedro")** a variável **nome** receberá **"Pedro"**, ou seja, o conteúdo da variável **nome** dependerá do valor passado como parâmetro:

```
function saudacao(nome){  
  console.log("Boa noite", nome)  
}  
  
//chamada da função saudacao  
saudacao("Ana")  
  
//chamada da função saudacao  
saudacao("Pedro")
```

```
Boa noite Ana  
Boa noite Pedro
```

IV. Função que recebe dois parâmetros e não tem retorno

A função **classificacao** recebe dois parâmetros e não tem retorno. Para chamar a função temos de colocar dois valores dentro dos parênteses separados por vírgula. Na chamada `classificacao("Ana",21)`, o valor **"Ana"** será colocado na variável **nome** e o valor **21** será colocado na variável **idade**. A ordem dos parâmetros é importante, a chamada `classificacao(21,"Ana")` não está correta, também não estará correto passar um número insuficiente de parâmetros:

```
function classificacao(nome,idade){  
  if( idade < 18 ){  
    console.log(nome, "é de menor")  
  }  
  else{  
    console.log(nome, "é de maior")  
  }  
}
```

```
//chamada da função classificacao  
classificacao("Ana",21)  
  
//chamada da função classificacao  
classificacao("Pedro",17)
```

```
Ana é de maior  
Pedro é de menor
```

Neste exemplo a função **classificacao** recebe dois parâmetros, mas uma função pode ser definida para receber qualquer quantidade de parâmetros.

V. Função que recebe dois parâmetros e tem retorno

A função **somar** recebe dois parâmetros e retorna a soma deles. Para chamar a função temos de colocar dois valores dentro dos parênteses separados por vírgula. Na chamada `somar(2,3)`, o valor **2** será colocado na variável **x** e o valor **3** será colocado na variável **y**. Como a função possui retorno, então é importante guardarmos o retorno numa variável, aqui o resultado foi colocado na variável **r** e, desta forma, foi possível usar ele na linha seguinte em `console.log("Resultado:", r)`:

```
function somar(x,y){
  return x + y
}

//chamada da função somar
//a variável r receberá o resultado
r = somar(2,3)
console.log("Resultado:", r)

//chamada da função somar
//a variável s receberá o resultado
s = somar(8,1)
console.log("Resultado:", s)
```

```
Resultado: 5
Resultado: 9
```

Exercícios

Para fazer os exercícios crie um projeto de nome **aula6** no VS Code assim como é mostrado a seguir. Cada programa deverá estar num arquivo separado da pasta **src**. Crie uma propriedade para cada exercício na propriedade **scripts** do arquivo **package.json**. Para rodar o arquivo use: **npm run propriedade**, onde propriedade será **um**, **dois**, ... e **oito**.

The screenshot shows a VS Code interface with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'AULA6' with a subfolder 'src' containing files 'cinco.js', 'dois.js', 'oito.js', 'quatro.js', 'seis.js', 'sete.js', 'tres.js', and 'um.js'. The code editor shows the 'package.json' file with the following content:

```
{
  "name": "aula6",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "um": "node ./src/um",
    "dois": "node ./src/dois",
    "tres": "node ./src/tres",
    "quatro": "node ./src/quatro",
    "cinco": "node ./src/cinco",
    "seis": "node ./src/seis",
    "sete": "node ./src/sete",
    "oito": "node ./src/oito"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

Exercício 1: Fazer uma função de nome **maior** que recebe dois números e retorna o maior deles. Ao lado tem-se

Exemplo de saída:

dois exemplos de chamadas dessa função.

```
m = maior(2, 3)
console.log("Maior:", m)

m = maior(5, 2)
console.log("Maior:", m)
```

Maior: 3
Maior: 5

Exercício 2: Fazer uma função de nome **primeira** que recebe uma string e retorna o 1º caractere dessa string. Ao lado tem-se dois exemplos de chamadas dessa função.

Exemplo de saída:

```
p = primeira("Pedro")
console.log("1a letra:", p)

p = primeira("Mariana")
console.log("1a letra:", p)
```

1a letra: P
1a letra: M

Exercício 3: Fazer uma função de nome **ultima** que recebe uma string e retorna o último caractere dessa string. Ao lado tem-se exemplos de chamadas dessa função.

Dica: use a propriedade `length` da string para obter o número de caracteres da string.

Exemplo de saída:

```
u = ultima("Pedro")
console.log("Última letra:", u)

u = ultima("Mariana")
console.log("Última letra:", u)
```

Última letra: o
Última letra: a

Exercício 4: Fazer uma função de nome **letra** que recebe uma string e um número inteiro indicando uma posição na string. A função retornará o caractere que está na posição fornecida como parâmetro. Ao lado tem-se exemplos de chamadas dessa função.

Exemplo de saída:

```
l = letra("Pedro", 1)
console.log("2a letra:", l)

l = letra("Mariana", 3)
console.log("4a letra:", l)
```

2a letra: e
4a letra: i

Exercício 5: Fazer uma função de nome **inverte** que recebe uma string e retorna ela invertida. Ao lado tem-se exemplos de chamadas dessa função.

Dicas:

- Use a propriedade `length` da string para obter o número de caracteres da string;
- Lembre-se que a última posição da string está uma

Exemplo de saída:

```
r = inverte("Pedro")
console.log("Invertido:", r)

r = inverte("Mariana")
console.log("Invertido:", r)
```

Invertido: ordeP
Invertido: aniraM

posição antes da quantidade de caracteres.

Exercício 6: Fazer uma função de nome **somatorio** que recebe um array e retorna o somatório dos elementos desse array. Ao lado tem-se exemplos de chamadas dessa função.

Dicas:

- Use a propriedade length do array para obter o número de elementos do array.

Exemplo de saída:

```
numeros = [8,3,4,7,5]
s = somatorio(numeros)
console.log("Somatório:", s)

numeros = [5,4,3]
s = somatorio(numeros)
console.log("Somatório:", s)
```

Somatório: 27
Somatório: 12

Exercício 7: Fazer uma função de nome **inverter** que recebe uma lista e imprime na tela os elementos da lista na ordem inversa. Ao lado tem-se exemplos de chamadas dessa função.

Dicas:

- Use a propriedade length do array para obter o número de elementos do array;
- Lembre-se que a última posição do array está uma posição antes da quantidade de elementos.

Exemplo de saída:

```
console.log("Lista:")
numeros = [8,3,4,7,5]
inverter(numeros)

console.log("\nLista:")
numeros = [5,4,3]
inverter(numeros)
```

Lista:
4 : 5
3 : 7
2 : 4
1 : 3
0 : 8

Lista:
2 : 3
1 : 4
0 : 5

Exercício 8: Fazer uma função de nome **matriz** que recebe uma matriz e retorna o somatório dos elementos. Ao lado tem-se exemplos de chamadas dessa função.

Dicas:

- Use duas estruturas de repetição while, uma para percorrer a 1ª dimensão do array e outro while para percorrer a 2ª dimensão do array.

Exemplo de saída:

```
numeros = [
  [9,2,4],
  [6,5,7],
  [2,1,3]
]
s = matriz(numeros)
console.log('Somatório:', s)

numeros = [
  [5,8,3,4],
  [9,7,2]
]
s = matriz(numeros)
console.log('Somatório:', s)
```

Somatório: 39
Somatório: 38