

**ESMAD | TSIW | POO**  
**Ficha de Exercícios nº2 – Funções**

Crie uma função para resolver cada uma destas alíneas. Para a codificação é aconselhado que use um playground Web:

**1. Declaração de função**

- a. Crie uma função que apresente numa caixa de alerta a frase “OLÁ MUNDO!”. Dê um bom nome à função. Execute a função 3 vezes

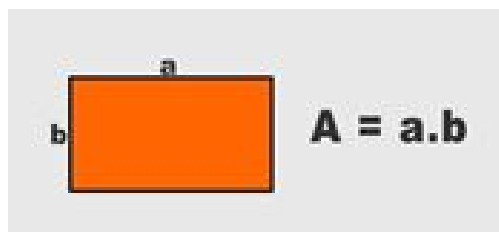
```
showMessage();  
showMessage();  
showMessage();  
  
function showMessage() {  
    alert("OLÁ MUNDO!");  
}
```

**2. Parâmetros**

- a. Crie uma função que imprima “OLÁ [name]!”, onde **name** é uma variável passada como um parâmetro. A variável **name** deve ser inicializada “hard-coded”

```
const name = "Rui";  
showName(name);  
  
function showName(name) {  
    alert(`OLÁ ${name}!`);  
}
```

- b. Crie uma função que imprima o resultado do cálculo da área de um retângulo passados por parâmetro o comprimento dos dois lados



```
calcRectArea(3, 4);
```

```
function calcRectArea(a, b) {
  alert(`A área do retângulo é ${a*b}`);
}
```

- c. Crie uma função que apresente numa caixa de alerta o resultado de operações aritméticas. A função deve receber dois valores inteiros e um operador (+, -, \*, /). Todos os valores devem ser previamente obtidos através de funções **prompt**. Salvar a divisão por 0 apresentando ao utilizador uma mensagem a informar que não é possível efetuar a divisão



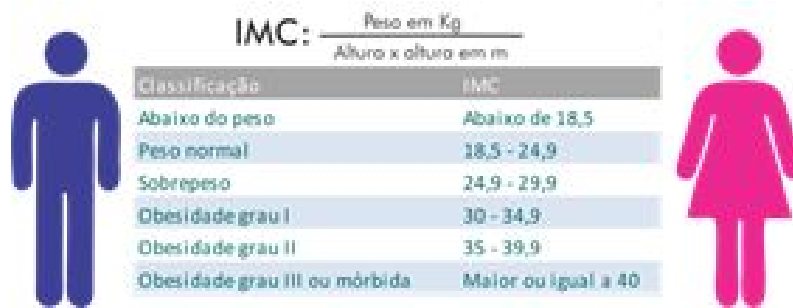
```
const num1 = +prompt("Digite o primeiro número?");
const num2 = +prompt("Digite o segundo número?");
const operator = prompt("Digite o operador?");
```

```
calculator(num1, num2, operator);
```

```
function calculator(num1, num2, operator) {
  switch (operator) {
    case '+':
      alert(`O resultado é ${num1 + num2}`);
      break;
    case '-':
      alert(`O resultado é ${num1 - num2}`);
      break;
    case '*':
      alert(`O resultado é ${num1 * num2}`);
      break;
    case '/':
      if (num2 === 0) {
        alert('Não é possível efetuar a divisão de um número por 0');
      } else {
        alert(`O resultado é ${num1 / num2}`);
      }
      break;
    default:
      alert('Operador inválido!')
  }
}
```

```
break;
}
}
```

- d. Crie uma função que apresente numa caixa de alerta a classificação do IMC (Índice de Massa Corporal) de uma pessoa recebendo como parâmetros o peso e a altura respetivos. Ambos os parâmetros devem ser recolhidos através de funções **prompt**



Classificação	IMC
Abaixo do peso	Abaixo de 18,5
Peso normal	18,5 - 24,9
Sobrepeso	24,9 - 29,9
Obesidade grau I	30 - 34,9
Obesidade grau II	35 - 39,9
Obesidade grau III, ou mórbida	Maior ou igual a 40

```
const weight = +prompt("Qual o seu peso?");
const height = +prompt("Qual a sua altura?");
```

```
calclmc(weight, height);
```

```
function calclmc(weight, height) {
  const imc = weight / (height * height);
  if (imc < 18.5) {
    alert("Abaixo do peso");
  } else if (imc >= 18.5 && imc <= 24.9) {
    alert("Peso normal");
  } else if (imc >= 25 && imc <= 29.9) {
    alert("Sobrepeso");
  } else if (imc >= 30 && imc <= 34.9) {
    alert("Obesidade grau I");
  } else if (imc >= 35 && imc <= 39.9) {
    alert("Obesidade grau II");
  } else {
    alert("Obesidade grau III");
  }
}
```

- e. Crie uma função que simule um eco. Dada uma string **s** e um número **n** deve imprimir **s** escrito **n** vezes

```
echo('ESMAD', 3);
```

```
function echo(s, n) {
  let result = "";
  for(let i = 0; i < n; i++) {
    result += s;
  }
}
```

```
}  
alert(result);  
}
```

- f. Crie uma função que comece por ler do utilizador dois valores inteiros **a** e **b**, e que escreva todos os valores inteiros pertencentes ao intervalo **[a,b[**

```
const a = +prompt("Digite o primeiro número?");  
const b = +prompt("Digite o segundo número?");  
  
showNumbers(a, b);  
  
function showNumbers(a, b) {  
    let result = "";  
    for(let i = a; i < b; i++) {  
        result += i;  
    }  
    alert(result);  
}
```

- g. Crie uma função que calcule a soma dos múltiplos de 3 existentes num intervalo **[a,b]**, em que **a** e **b** são passados como parâmetros

```
const a = +prompt("Digite o primeiro número?");  
const b = +prompt("Digite o segundo número?");  
  
showMultiples(a, b);  
  
function showMultiples(a, b) {  
    let result = 0;  
    for(let i = a; i <= b; i++) {  
        if(i % 3 === 0)  
            result += i;  
    }  
    alert(result);  
}
```

- h. Crie uma função que devolva a tabuada de um determinado número passado como parâmetro. Se nenhum número for passado a função deve imprimir a tabuada do 1

1 X 1 = 1 1 X 2 = 2 1 X 3 = 3 1 X 4 = 4 1 X 5 = 5 1 X 6 = 6 1 X 7 = 7 1 X 8 = 8 1 X 9 = 9 1 X 10 = 10	2 X 1 = 2 2 X 2 = 4 2 X 3 = 6 2 X 4 = 8 2 X 5 = 10 2 X 6 = 12 2 X 7 = 14 2 X 8 = 16 2 X 9 = 18 2 X 10 = 20	3 X 1 = 3 3 X 2 = 6 3 X 3 = 9 3 X 4 = 12 3 X 5 = 15 3 X 6 = 18 3 X 7 = 21 3 X 8 = 24 3 X 9 = 27 3 X 10 = 30	4 X 1 = 4 4 X 2 = 8 4 X 3 = 12 4 X 4 = 16 4 X 5 = 20 4 X 6 = 24 4 X 7 = 28 4 X 8 = 32 4 X 9 = 36 4 X 10 = 40	5 X 1 = 5 5 X 2 = 10 5 X 3 = 15 5 X 4 = 20 5 X 5 = 25 5 X 6 = 30 5 X 7 = 35 5 X 8 = 40 5 X 9 = 45 5 X 10 = 50
6 X 1 = 6 6 X 2 = 12 6 X 3 = 18 6 X 4 = 24 6 X 5 = 30 6 X 6 = 36 6 X 7 = 42 6 X 8 = 48 6 X 9 = 54 6 X 10 = 60	7 X 1 = 7 7 X 2 = 14 7 X 3 = 21 7 X 4 = 28 7 X 5 = 35 7 X 6 = 42 7 X 7 = 49 7 X 8 = 56 7 X 9 = 63 7 X 10 = 70	8 X 1 = 8 8 X 2 = 16 8 X 3 = 24 8 X 4 = 32 8 X 5 = 40 8 X 6 = 48 8 X 7 = 56 8 X 8 = 64 8 X 9 = 72 8 X 10 = 80	9 X 1 = 9 9 X 2 = 18 9 X 3 = 27 9 X 4 = 36 9 X 5 = 45 9 X 6 = 54 9 X 7 = 63 9 X 8 = 72 9 X 9 = 81 9 X 10 = 90	10 X 1 = 10 10 X 2 = 20 10 X 3 = 30 10 X 4 = 40 10 X 5 = 50 10 X 6 = 60 10 X 7 = 70 10 X 8 = 80 10 X 9 = 90 10 X 10 = 100

```
const num = +prompt("Digite um número?");
```

```
showMultiplicationNumber(num);
```

```
function showMultiplicationNumber(num = 1) {
    let result = "";
    for(let i = 1; i <= 10; i++) {
        result += `${num} x ${i} = ${num*i}\n`;
    }
    alert(result);
}
```

- i. Crie uma função que soma **N** números passado por parâmetros, Use o objeto **arguments** para resolver este problema

```
sumNums(3, 4);
sumNums(7, 12, 9, 1);
```

```
function sumNums() {
    let result = 0;
    for(let i = 0; i < arguments.length; i++) {
        result += arguments[i];
    }
    alert(result);
}
```

- j. Crie uma função que receba o primeiro e último nome de uma criança e um conjunto de strings que representam o nome de cada uma das suas amigas. A função deve apresentar a seguinte frase: **"A [primeiroNome] [ultimoNome] tem [nAmigas] amigas!"**. Use o parâmetro **Rest** para resolver esta questão

```
showFriends("Maria", "Silva");
showFriends("Maria", "Silva", "Inês");
showFriends("Maria", "Silva", "Joana", "Rute", "Inês");
```

```
function showFriends(firstName, lastName, ...friends) {
```

```

alert(`A ${firstName} ${lastName} tem ${friends.length} amigas!`);
}

```

### 3. Retorno de valor

- a. Crie uma função **min(a, b)** que retorna o mínimo de dois números **a** e **b**.  
Casos de teste:

```

min(2, 5) == 2
min(3, -1) == -1
min(1, 1) == 1

```

```

alert(min(2, 5));
alert(min(3, -1));
alert(min(1, 1));

function min(num1, num2) {
    if(num1 < num2) {
        return num1;
    } else {
        return num2;
    }
}

```

- b. Crie uma função **pow(x, n)** que retorne **x** na potência **n**. Ou, em outras palavras, multiplica **x** por si **n** vezes e retorna o resultado.  
Casos de teste:

```

pow(3, 2) = 3 * 3 = 9
pow(3, 3) = 3 * 3 * 3 = 27
pow(1, 100) = 1 * 1 * ... * 1 = 1

```

Antes de chamar a função, deve solicitar **x** e **n** ao utilizador. A função deve suportar apenas valores naturais de **n**: inteiros acima de 1

```

alert(pow(3, 2));
alert(pow(3, 3));
alert(pow(4, 'x'));
alert(pow(1, 100));

function pow(base, exponent) {
    if(isNaN(+base) || base <= 1) {
        return "Base deve ser um número inteiro superior a 1";
    } else if(isNaN(+exponent) || exponent <= 1) {
        return "Expoente deve ser um número inteiro superior a 1";
    } else {
        return base ** exponent;
    }
}

```

```
}
```

- c. Crie uma função que verifique se um número passado como parâmetro é ou não primo. Relembre-se que um número primo **N** é um número natural maior do que 1 que não possui divisores além de 1 e de si mesmo. Retorne true caso seja primo e false, caso contrário. A função deve ser invocada dentro de uma estrutura **IF** e caso seja verdadeiro deve ser apresentado “O número **[N]** é primo”. Caso contrário, “O número **[N]** não é primo”

```
const num = 8;

if(checkPrime(num)) {
    alert(`O número ${num} é primo!`)
} else {
    alert(`O número ${num} não é primo!`)
}

function checkPrime(num) {
    let isPrime = true;
    for(let i = 2; i < num; i++) {
        if(num % i === 0) {
            isPrime = false;
            break;
        }
    }
    return isPrime;
}
```

- d. Crie uma função que retorne o fatorial de um valor inteiro positivo passado como parâmetro. Exemplos:  $0!=1$ ;  $1!=1$ ;  $5!=5 \times 4 \times 3 \times 2 \times 1 = 120$ ;  $6!=6 \times 5 \times 4 \times 3 \times 2 \times 1 = 720$ ;

```
alert(calcFactorial(0));
alert(calcFactorial(1));
alert(calcFactorial(5));
alert(calcFactorial(6));

function calcFactorial(num) {
    let sum = 1;
    for(let i = 1; i <= num; i++) {
        sum *= i
    }
    return sum;
}
```

#### 4. Expressão de função

- a. Crie uma expressão de função chamada **isLeapYear** que deve retornar **true** se um ano passado pelo utilizador é bissexto e **false** caso contrário.



```
let checkLeapYear = function(year) {  
  return ((year % 4 == 0) && (year % 100 != 0)) || (year % 400 == 0);  
}
```

```
alert(checkLeapYear(2000));  
alert(checkLeapYear(2019));  
alert(checkLeapYear(2016));
```

- b. Crie uma expressão função chamada **isPerfect** que verifique se um número é perfeito. Um número perfeito é um número natural para o qual a soma de todos os seus divisores naturais próprios (excluindo ele mesmo) é igual ao próprio número. Por exemplo, o número 28 é, pois:  $28=1+2+4+7+14$ . Se for perfeito deve retornar **true**. Caso contrário, deve retornar **false**. Use a função num ciclo que deve continuar a pedir um número ao utilizador até que o número digitado seja perfeito.

```
let checkPerfect = function(num) {  
  let temp = 0;  
  for(let i =1; i <= num/2; i++) {  
    if(num % i === 0) {  
      temp += i;  
    }  
  }  
  if(temp === num && temp !== 0) {  
    return true;  
  } else {  
    return false;  
  }  
}
```



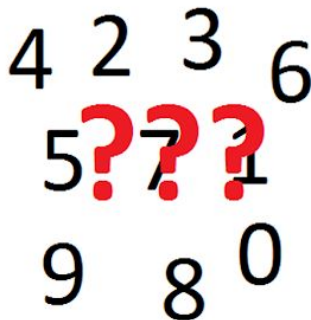
```
let result;
do {
    const num = +prompt("Numero: ");
    result = checkPerfect(num);
}while(!result);
```

- c. Crie uma expressão de função abreviada (função arrow) que dado um número de 100 a 999 verifique se o mesmo é um número palíndromo (capicua). Caso seja, deve devolver **true**. Caso contrário, deve devolver **false**. Escolha um bom nome para a função.

```
let checkPalindrome = num => Math.floor(num/100) === num % 10

alert(checkPalindrome(121));
alert(checkPalindrome(283));
alert(checkPalindrome(350));
```

5. Faça o jogo da adivinha. Comece por gerar um número aleatório entre 1 e 100. Depois vá perguntando ao utilizador para adivinhar o número. Caso o utilizador digite um número superior deve exibir o seguinte texto **"PARA BAIXO"**. Caso contrário, deve indicar: **"PARA CIMA"**. Caso acerte, deve exibir a mensagem **"ADIVINHOU, PARABÉNS!"**.



Neste jogo deve ter uma função que recebe dois parâmetros: o número a adivinhar e a tentativa do utilizador. A função deve retornar:

- 1 se a tentativa for INFERIOR ao número gerado inicialmente
- 1 se a tentativa for SUPERIOR ao número gerado inicialmente
- 0 se a tentativa for IGUAL ao número gerado inicialmente

Torne o jogo mais interessante, e forneça apenas 5 tentativas ao jogador. Caso atinja o limite, deve indicar: **"PACIÊNCIA, JOGUE OUTRA VEZ!"**.

```
let attempts = 0;
let result;
const limitAttempts = 5;
const randomNumber = Math.floor((Math.random() * 100) + 1);

do {
    attempts++;
    const attemptNumber = +prompt("Digite um número?");
    result = checkNumber(randomNumber, attemptNumber);
    if(result === 0) {
        break;
    } else if(result > 0) {
        alert("PARA BAIXO!");
    } else {
        alert("PARA CIMA!");
    }
} while(attempts < limitAttempts)

if(result === 0) {
    alert("ADIVINHOU, PARABÉNS!");
} else {
    alert("PACIÊNCIA, JOGUE OUTRA VEZ");
}

function checkNumber(randomNumber, attemptNumber) {
    if(randomNumber === attemptNumber) {
        return 0;
    } else if(randomNumber > attemptNumber) {
        return -1;
    } else {
        return 1;
    }
}
```