ESMAD | TSIW | POO

Ficha de Exercícios nº1 – Introdução ao JavaScript

Use um playground (ex.: CodePen, Jsfiddle, etc.) ou use o Chrome DevTools para resolver aos seguintes exercícios:

1. Crie uma instrução que exiba a mensagem "Olá JavaScript!".

```
alert( "Olá JavaScript!" );
```

2. Variáveis

a. Trabalhar com variáveis
   i. Declare duas variáveis: admin e name.
   ii. Atribua o valor "John" à variável name.
   iii. Copie o valor do name para admin.
   iv. Mostre o valor de admin usando uma caixa de alerta (deve ser exibido "John").

```
let admin, name;
name = "John";
admin = name;
alert(admin); // "John"
```

b. Nomeação de variáveis:
   i. Crie uma variável com o nome do nosso planeta. Como chamaria a essa variável?

```
let ourPlanetName = "Terra";
```

   ii. Crie uma variável para armazenar o nome de um visitante atual de um site. Como nomearia essa variável?

```
let currentUserName = "John";
```

3. Constantes:

a. Examine o seguinte código que inclui uma data de aniversário constante e a idade é calculada a partir do aniversário com a ajuda de algum código (não é fornecido por questões de foco):
   const birthday = '18 .04.1982 ';
   const age = someCode (birthday);

b. Seria correto usar letras maiúsculas para a variável **birthday**? E para **age**? Ou até para as duas?
const BIRTHDAY = '18 .04.1982 '; // maiúsculas?
const AGE = someCode (BIRTHDAY); // maiúsculas?

```
const BIRTHDAY = '18.04.1982';
const age = someCode(BIRTHDAY);
```

4. Tipos de dados:

a. Qual o output da seguinte script:
let name = "Ilya";
alert(`hello ${1}`); // ?
alert(`hello ${"name"}`); // ?
alert(`hello ${name}`); // ?

```
let name = "Ilya";

alert( `hello ${1}` ); // hello 1
alert( `hello ${"name"}` ); // hello name
alert( `hello ${name}` ); // hello Ilya
```

5. Conversões de tipos:

a. Quais são os resultados dessas expressões:
"" + 1 + 0
"" - 1 + 0
true + false
6 / "3"
"2" * "3"
4 + 5 + "px"
"$" + 4 + 5
"4" - 2
"4px" - 2
7 / 0
" -9 " + 5
" -9 " - 5
null + 1
undefined + 1

```
"" + 1 + 0 = "10" // (1)
"" - 1 + 0 = -1 // (2)
true + false = 1
```

```
6 / "3" = 2
"2" * "3" = 6
4 + 5 + "px" = "9px"
"$" + 4 + 5 = "$45"
"4" - 2 = 2
"4px" - 2 = NaN
7 / 0 = Infinity
" -9  " + 5 = " -9  5" // (3)
" -9  " - 5 = -14 // (4)
null + 1 = 1 // (5)
undefined + 1 = NaN // (6)
```

1. A adição com uma string "" + 1 converte 1 numa string: "" + 1 = "1", e então nós temos "1" + 0, a mesma regra é aplicada.
2. A subtração - (como a maioria das operações matemáticas) só funciona com números, converte uma string vazia "" em 0.
3. A adição com uma string anexa o número 5 à string.
4. A subtração sempre se converte em números, portanto, torna "-9" um número -9 (ignorando espaços ao redor).
5. null torna-se 0 após a conversão numérica.
6. undefined torna-se NaN após a conversão numérica.

6. Operadores:

a. Quais são os valores finais de todas as variáveis a, b, ced após o código abaixo?

```
let a = 1, b = 1;

let c = ++a; // ?
let d = b++; // ?
```

Respostas:

- a = 2
- b = 2
- c = 2
- d = 1

b. Quais são os valores de **a** e **x** depois do código abaixo?

```
let a = 2;
let x = 1 + (a *= 2);
```

- `a = 4` (multiplicado por 2)
- `x = 5` (calculado como 1 + 4)

7. Interação:

a. Crie uma instrução que solicite um nome e imprima o nome juntando asteriscos antes e depois do nome.

```
let name = prompt("Qual o seu nome?", "");
alert(`*${name}*`);
```

8. Comparações:

a. Qual será o resultado dessas expressões?

5 > 4
"apple" > "pineapple"
"2" > "12"
undefined == null
undefined === null
null == "\n0\n"
null === +"\n0\n"

```
5 > 4 → true
"apple" > "pineapple" → false
"2" > "12" → true
undefined == null → true
undefined === null → false
null == "\n0\n" → false
null === +"\n0\n" → false
```

1. Obviamente, é verdade.
2. Comparação de dicionário, portanto, falsa.
3. Novamente, comparação de dicionário, primeiro caractere de "2" é maior que o primeiro caractere de "1".
4. Valores nulos e indefinidos são iguais entre si.
5. A igualdade estrita é rigorosa. Diferentes tipos de ambos os lados levam a falso.
6. Veja (4).
7. Igualdade estrita de diferentes tipos.

9. Condicionais:

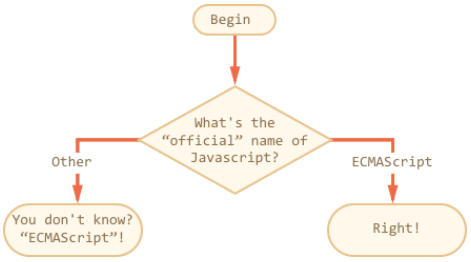
a. O alerta será mostrado?

```
if ("0") {
  alert( 'Hello' );
}
```

Sim, será mostrado.

Qualquer string, exceto uma vazia (e "0" não está vazia) torna-se verdadeira no contexto lógico.

b. Usando a construção if..else, escreva o código que pergunta: "What's the "official" name of Javascript?". Se o visitante digitar "ECMAScript", envie "Right!", Caso contrário, a saída: "You don't know? "ECMAScript"!"




```
let value = prompt('What is the "official" name of JavaScript?', '');

if (value == 'ECMAScript') {
   alert('Right!');
} else {
   alert("You don't know? ECMAScript!");
}
```

c. Usando if..else, escreva o código que obtém um número via prompt e, em seguida, mostra em alerta:
   i. 1, se o valor for maior que zero,
   ii. -1, se menor que zero,
   iii. 0, se igual a zero.

   Nesta tarefa, assumimos que a entrada é sempre um número.

```
let value = prompt('Escreva um número', 0);

if (value > 0) {
  alert( 1 );
} else if (value < 0) {
  alert( -1 );
} else {
  alert( 0 );
}
```

d. Reescreva este if com o operador ternário '?':

```
if (a + b < 4) {
  result = 'Below';
} else {
  result = 'Over';
}
```

```
result = (a + b < 4) ? 'Below' : 'Over';
```

e. Reescreva o próximo if...else usando múltiplos operadores ternários '?'.Para facilitar a leitura, é recomendável dividir o código em várias linhas.

```
let message;

if (login == 'Employee') {
  message = 'Hello';
} else if (login == 'Director') {
  message = 'Greetings';
} else if (login == '') {
  message = 'No login';
} else {
  message = '';
}
```

```
let message = (login == 'Employee') ? 'Hello' :
  (login == 'Director') ? 'Greetings' :
  (login == '') ? 'No login' :
  '';
```

10. Operadores lógicos:

a. Qual o output para cada linha de código?

```
alert( null || 2 || undefined );
alert( alert(1) || 2 || alert(3) );
alert( 1 && null && 2 );
alert( alert(1) && alert(2) );
alert( null || 2 && 3 || 4 );
```

alert(null || 2 || undefined);
A resposta é 2, esse é o primeiro valor verdadeiro.

alert(alert(1) || 2 || alert(3));
A resposta: primeiro 1 e depois 2.
A chamada para alert não retorna um valor, ou melhor, retorna undefined.
O primeiro OR || avalia o alerta esquerdo (1). Isso mostra a primeira mensagem com 1.
O alerta retorna indefinido, então OU passa para o segundo operando procurando por um valor verdadeiro. O segundo operando 2 é verdadeiro, então a execução é interrompida, 2 é retornado e então mostrado pelo alerta externo. Não haverá 3, porque a avaliação não alcança o alerta (3).

alert(1 && null && 2);
A resposta: null, porque é o primeiro valor falso da lista.

alert(alert(1) && alert(2));
A resposta: 1 e, em seguida, undefined.
A chamada para alerta retorna undefined (apenas mostra uma mensagem, portanto, não há retorno significativo). Por causa disso, && avalia o operando esquerdo (mostra 1), e imediatamente pára, porque undefined é um valor falso. E && procura por um valor falso e retorna.

alert(null || 2 && 3 || 4)
Resposta: 3
A precedência de && é maior que ||, portanto, é executada primeiro.
O resultado de 2 && 3 = 3, então a expressão torna-se:

null || 3 || 4

Agora o resultado é o primeiro valor verdadeiro (truthy): 3.

b. Escreva uma condição "if" para verificar se a idade está entre 14 e 90, inclusive. "Inclusive" significa que a idade pode atingir as extremidades 14 ou 90.

```
if (age >= 14 && age <= 90)
```

c. Escreva uma condição if para verificar se a idade NÃO está entre 14 e 90, inclusive. Crie duas variantes: a primeira usando NOT !, a segunda sem ela.

```
//1ª variante
if (!(age >= 14 && age <= 90))

//2ª variante
if (age < 14 || age > 90)
```

d. Quais desses alertas serão executados? Quais serão os resultados das expressões dentro if (...)?
if (-1 || 0) alert('first');
if (-1 && 0) alert('second');
if (null || -1 && 1) alert('third');

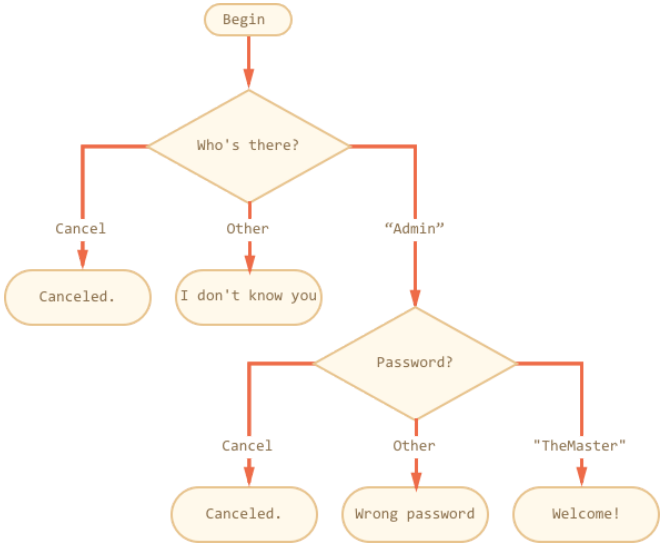
A resposta: o primeiro e o terceiro serão executados.

// Executa.
// O resultado de -1 || 0 = -1, truthy
if (-1 || 0) alerta (first);

```
// não executa
// -1 && 0 = 0, falsamente
if (-1 && 0) alerta (second);

// Executa
// Operador && tem uma precedência maior que ||
// então -1 && 1 executa primeiro, nos dando a expresão:
// null || -1 && 1 -> null || 1 -> 1
if (null || -1 && 1) alert ('third');
```

e. Escreva o código que pede um login com prompt.
   i. Se o visitante digitar "Admin", solicite uma senha, se a entrada for uma linha vazia ou Esc - exibir "Cancelado". Se for outra, mostre "Não conheço você".
   ii. A senha é verificada da seguinte forma:
      1. Se for igual a “TheMaster”, então mostre “Welcome!”,
      2. Outra string - mostra "Senha incorreta",
      3. Para uma sequência vazia ou entrada cancelada, mostre "Cancelado".
   iii. O esquema:



Por favor, use blocos aninhados. Observe a legibilidade geral do código.

```
let userName = prompt("Who's there?", '');

if (userName == 'Admin') {

  let pass = prompt('Password?', '');

  if (pass == 'TheMaster') {
    alert( 'Welcome!' );
  } else if (pass == '' || pass == null) {
```

```
    alert( 'Canceled.' );
  } else {
    alert( 'Wrong password' );
  }

} else if (userName == '' || userName == null) {
  alert( 'Canceled' );
} else {
  alert( "I don't know you" );
}
```

11. Ciclos:

a. Qual é o último valor alertado por este código? Por quê?

```
let i = 3;

while (i) {
  alert( i-- );
}
```

Resposta: 1

Cada iteração de loop diminui i por 1. A verificação while(i) interrompe o loop quando i = 0.

Portanto, as etapas do loop formam a seguinte sequência:

let i = 3;

alerta (i--); // mostra 3, diminui i para 2

alerta (i--) // mostra 2, diminui i para 1

alerta (i--) // mostra 1, diminui i para 0

// feito, i passa a 0 e a condição é avaliada como falsa

b. Para cada iteração de loop, anote o valor que ele gera e, em seguida, compare-o com a solução. Ambos os loops alertam os mesmos valores ou não?

i. The prefix form ++i:

```
let i = 0;
while (++i < 5) alert( i );
```

ii. The postfix form ++i:

```
let i = 0;
while (i++ < 5) alert( i );
```

Não!
Primeiro código o resultado é: 1 2 3 4
Segundo código o resultado é: 1 2 3 4 5

c. Para cada loop anote quais valores ele irá mostrar. Em seguida, compare com a resposta. Ambos os loops alertam os mesmos valores ou não?

i. The prefix form ++i:
for (let i = 0; i < 5; i++) alert(i);

ii. The postfix form ++i:
for (let i = 0; i < 5; ++i) alert(i);

A resposta: de 0 a 4 em ambos os casos.

d. Use o loop for para gerar números pares de 2 a 10.

```
for (let i = 2; i <= 10; i++) {
  if (i % 2 == 0) {
    alert( i );
  }
}
```

e. Reescreva o código alterando o loop for sem alterar seu comportamento (a saída deve permanecer igual).
for (let i = 0; i < 3; i++) {
alert(`number ${i}!`);
}

```
let i = 0;
while (i < 3) {
  alert( `number ${i}!` );
  i++;
}
```

f. Escreva um loop que solicite um número maior que 100. Se o visitante inserir outro número, peça-lhes para inserir novamente. O loop deve solicitar um número até que o visitante insira um número maior que 100 ou cancele a entrada / insira uma linha vazia. Aqui podemos supor que o visitante só insere números. Não há necessidade de implementar um tratamento especial para uma entrada não numérica nessa tarefa.

```
let num;

do {
  num = prompt("Enter a number greater than 100?", 0);
} while (num <= 100 && num);
```

g. Um número inteiro maior que 1 é chamado de primo se não puder ser dividido sem um resto por qualquer coisa exceto 1 e ele mesmo. Em outras palavras, n> 1 é primo se não puder ser dividido por qualquer coisa, exceto 1 e n. Por exemplo, 5 é um primo, porque não pode ser dividido sem um resto por 2, 3 e 4.
Escreva o código que gera números primos no intervalo de 2 a n.
Para n = 10, o resultado será 2,3,5,7.
P.S. O código deve funcionar para qualquer n, não deve ser ajustado para qualquer valor fixo.

```
let n = 10;

nextPrime:
for (let i = 2; i <= n; i++) {

  for (let j = 2; j < i; j++) {
    if (i % j == 0) continue nextPrime;
  }

  alert( i ); // a prime
}
```

12. Switch

a. Escreva o código usando if..else que corresponderia ao seguinte switch:

```
switch (browser) {
  case 'Edge':
    alert( "You've got the Edge!" );
    break;

  case 'Chrome':
  case 'Firefox':
  case 'Safari':
  case 'Opera':
    alert( 'Okay we support these browsers too' );
    break;

  default:
    alert( 'We hope that this page looks ok!' );
}
```

```
if(browser == 'Edge') {
  alert("You've got the Edge!");
} else if (browser == 'Chrome'
 || browser == 'Firefox'
 || browser == 'Safari'
 || browser == 'Opera') {
  alert( 'Okay we support these browsers too' );
} else {
  alert( 'We hope that this page looks ok!' );
}
```

b. Reescreva o código abaixo usando uma única instrução switch:

```
let a = +prompt('a?', '');

if (a == 0) {
  alert( 0 );
}
if (a == 1) {
  alert( 1 );
}
if (a == 2 || a == 3) {
  alert( '2,3' );
}
```

```
let a = +prompt('a?', '');

switch (a) {
  case 0:
    alert( 0 );
    break;

  case 1:
    alert( 1 );
    break;

  case 2:
  case 3:
    alert( '2,3' );
    break;
}
```