Exercícios de JavaScript - Gabriel Silva

Conteúdo: Destructuring, Spread/Rest, Rest Parameters, Short Circuiting, Logical Assignment Operators

Destructuring

1. 1. Crie uma função que receba um array com três elementos e retorne o segundo usando destructuring.
   Solução:
   (escreva aqui)

2. 2. Separe o primeiro e o restante dos elementos de um array usando destructuring e rest.
   Solução:
   (escreva aqui)

3. 3. Extraia dois valores de um objeto e renomeie-os no processo.
   Solução:
   (escreva aqui)

4. 4. Destructure um array aninhado com dois níveis de profundidade.
   Solução:
   (escreva aqui)

5. 5. Utilize destructuring para trocar os valores de duas variáveis.
   Solução:
   (escreva aqui)

6. 6. Destructure um objeto e defina valores padrão para as chaves.
   Solução:
   (escreva aqui)

7. 7. Use destructuring para extrair dados de um objeto passado como argumento de função.
   Solução:
   (escreva aqui)

8. 8. Combine destructuring com um loop para imprimir valores de um array de objetos.
   Solução:

(escreva aqui)

9. 9. Destructure parâmetros diretamente na assinatura da função.
   Solução:
   (escreva aqui)

10. 10. Destructure uma tupla representada por array para somar seus dois números.
    Solução:
    (escreva aqui)

Spread / Rest

11. 1. Utilize spread para clonar um array e adicione novos elementos.
    Solução:
    (escreva aqui)

12. 2. Combine dois arrays usando spread.
    Solução:
    (escreva aqui)

13. 3. Use spread para passar um array como argumento de uma função que usa múltiplos parâmetros.
    Solução:
    (escreva aqui)

14. 4. Use rest para capturar argumentos excedentes de uma função.
    Solução:
    (escreva aqui)

15. 5. Crie uma função que soma todos os números passados usando rest.
    Solução:
    (escreva aqui)

16. 6. Use spread para clonar um objeto e modificar uma propriedade.
    Solução:
    (escreva aqui)

17. 7. Combine objetos com spread, onde o segundo sobrescreve propriedades do primeiro.
    Solução:
    (escreva aqui)

18. 8. Use rest para separar uma propriedade de um objeto e agrupar o restante.
    Solução:
    (escreva aqui)

19. 9. Use spread para criar um novo array com um valor no início e outro no fim.
    Solução:
    (escreva aqui)

20. 10. Use rest no destructuring de arrays dentro de um loop.
    Solução:
    (escreva aqui)

Rest Parameters

21. 1. Crie uma função que recebe qualquer quantidade de argumentos e imprime a média.
    Solução:
    (escreva aqui)

22. 2. Implemente uma função que concatene qualquer número de strings.
    Solução:
    (escreva aqui)

23. 3. Utilize rest parameters para criar uma função que retorna o maior valor passado.
    Solução:
    (escreva aqui)

24. 4. Crie uma função que aceita qualquer número de argumentos e conta quantos são strings.
    Solução:
    (escreva aqui)

25. 5. Crie uma função que calcula o produto de todos os números passados.
    Solução:
    (escreva aqui)

26. 6. Crie uma função que aceita nomes e imprime uma saudação para cada um.
    Solução:
    (escreva aqui)

27. 7. Implemente uma função que verifica quantos argumentos são maiores que 10.
    Solução:

(escreva aqui)

28. 8. Crie uma função que conta quantos argumentos são do tipo boolean.
    Solução:
    (escreva aqui)

29. 9. Crie uma função que soma apenas os números pares passados como argumentos.
    Solução:
    (escreva aqui)

30. 10. Crie uma função que retorna um array com os tipos dos argumentos passados.
    Solução:
    (escreva aqui)

Short Circuiting

31. 1. Use `||` para definir um valor padrão para uma variável.
    Solução:
    (escreva aqui)

32. 2. Use `&&` para executar uma função somente se uma condição for verdadeira.
    Solução:
    (escreva aqui)

33. 3. Use `||` para simular um fallback de configuração.
    Solução:
    (escreva aqui)

34. 4. Utilize `??` para diferenciar `null`/`undefined` de falsy values como 0.
    Solução:
    (escreva aqui)

35. 5. Combine `&&` com funções para evitar chamadas desnecessárias.
    Solução:
    (escreva aqui)

36. 6. Use `||` para inicializar um valor com base em outra variável.
    Solução:
    (escreva aqui)

37. 7. Use `&&` para evitar erro ao acessar propriedade de objeto possivelmente undefined.
    Solução:

(escreva aqui)

38. 8. Combine `||` e `&&` para criar um fallback condicional.
Solução:
(escreva aqui)

39. 9. Crie uma cadeia de `||` para encontrar o primeiro valor válido entre variáveis.
Solução:
(escreva aqui)

40. 10. Use `??` para aceitar 0 ou string vazia mas não null/undefined.
Solução:
(escreva aqui)

Logical Assignment Operators

41. 1. Use ||= para definir uma propriedade default em um objeto.
Solução:
(escreva aqui)

42. 2. Use &&= para redefinir um valor se ele for truthy.
Solução:
(escreva aqui)

43. 3. Use ??= para definir um valor somente se for null ou undefined.
Solução:
(escreva aqui)

44. 4. Crie um loop que usa ||= para inicializar objetos dentro de um array.
Solução:
(escreva aqui)

45. 5. Utilize &&= para executar uma atualização apenas se uma flag for true.
Solução:
(escreva aqui)

46. 6. Use ??= para aplicar uma configuração padrão a uma propriedade ausente.
Solução:
(escreva aqui)

47. 7. Crie uma função que usa ||= para preencher campos vazios de um usuário.
Solução:

(escreva aqui)

48. 8. Aplique &&= para limpar um cache somente se ele estiver habilitado.
    Solução:
    (escreva aqui)

49. 9. Use ??= para inicializar variáveis não definidas com objetos.
    Solução:
    (escreva aqui)

50. 10. Implemente um contador que só inicia com valor padrão se não for definido.
    Solução:
    (escreva aqui)

Destructuring

51. 1. const [, second] = [1, 2, 3];
52. 2. const [first, ...rest] = [10, 20, 30, 40];
53. 3. const {name: nome, age: idade} = {name: 'Ana', age: 25};
54. 4. const [[a], [b]] = [[1], [2]];
55. 5. [a, b] = [b, a];
56. 6. const {x = 10, y = 5} = {};
57. 7. function showUser({name, age}) { console.log(name, age); }
58. 8. for (const {id, value} of arr) { console.log(id, value); }
59. 9. function print({name, score}) { console.log(name, score); }
60. 10. const [x, y] = [5, 10]; const sum = x + y;

Spread / Rest

61. 1. const clone = [...arr, 'new'];
62. 2. const combo = [...arr1, ...arr2];
63. 3. Math.max(...[10, 20, 30]);
64. 4. function sumAll(...nums) { return nums.reduce((a, b) => a + b); }
65. 5. function sum(...nums) { return nums.reduce((a, b) => a + b, 0); }
66. 6. const user2 = {...user, age: 30};
67. 7. const merged = {...obj1, ...obj2};
68. 8. const {a, ...rest} = {a:1, b:2, c:3};
69. 9. const arr = [1, ...mid, 5];
70. 10. for (const [first, ...rest] of list) console.log(rest);

Rest Parameters

71. 1. function average(...nums) { return nums.reduce((a,b)=>a+b)/nums.length; }
72. 2. function concatAll(...strings) { return strings.join(''); }
73. 3. function max(...nums) { return Math.max(...nums); }
74. 4. function countStrings(...args) { return args.filter(a => typeof a === 'string').length; }
75. 5. function product(...nums) { return nums.reduce((a,b)=>a*b,1); }
76. 6. function greetAll(...names) { names.forEach(n => console.log(`Hi ${n}`)); }
77. 7. function countAbove10(...nums) { return nums.filter(n => n > 10).length; }
78. 8. function countBooleans(...args) { return args.filter(a => typeof a === 'boolean').length; }
79. 9. function sumEven(...nums) { return nums.filter(n => n%2===0).reduce((a,b)=>a+b,0); }
80. 10. function types(...args) { return args.map(a => typeof a); }

Short Circuiting

81. 1. let name = input || 'Guest';
82. 2. isLoggedIn && showDashboard();
83. 3. config.port = config.port || 3000;
84. 4. let count = val ?? 0;
85. 5. isReady && doSomething();
86. 6. let level = user.level || 'basic';
87. 7. const name = user && user.info && user.info.name;
88. 8. (config.retry || config.defaultRetry) && tryAgain();
89. 9. const valid = a || b || c || d;
90. 10. let name = input ?? 'Anonymous';

Logical Assignment Operators

91. 1. obj.name ||= 'Default';
92. 2. flag &&= false;
93. 3. value ??= 10;
94. 4. for (let i of arr) i.obj ||= {};
95. 5. isEnabled &&= updateData();
96. 6. settings.theme ??= 'light';
97. 7. user.name ||= 'Anonymous';
98. 8. cache.enabled &&= clearCache();
99. 9. options.config ??= {};
100. 10. counter ??= 1;