

Exercícios de Lógica de Programação em JavaScript: Aprimore suas Habilidades

[Início](#) [O que é javascript](#) [Variáveis](#) [Operadores](#) [Tipos de Dados](#)

[Exercícios de lógica](#) [if else switch](#) [Ternário](#)



👤 Eduardo Henrique Gomes 📅 14/08/2023 ⌚ 10:46

Tabela de Conteúdo [[Mostrar](#)]

Imprimir PDF

Ver PDF

SmartConvertPDF

Open

Introdução

Se você está procurando melhorar suas habilidades em programação e está focado em JavaScript, você veio ao lugar certo! Neste artigo, exploraremos uma variedade de exercícios de lógica de programação em JavaScript, projetados para aprimorar sua capacidade de resolver problemas e pensar de maneira algorítmica. Vamos mergulhar nos detalhes e aprofundar nossos conhecimentos!

Conceitos Básicos de Lógica de Programação

Antes de mergulharmos nos exercícios práticos, é crucial entender os conceitos básicos de lógica de programação. Vamos explorar termos como variáveis, condicionais, loops e funções.

Variáveis e Tipos de Dados

Começaremos com o básico: variáveis e tipos de dados. Em JavaScript, as variáveis são usadas para armazenar informações. Elas podem conter diferentes tipos de dados, como números, strings e booleanos.

Condicionais e Declarações de Controle

As estruturas condicionais, como `if`, `else if` e `else`, são fundamentais para tomar decisões em um programa. Vamos entender como executar diferentes blocos de código com base em condições específicas.

Exemplo 1: Utilizando `if`, `else if` e `else`

```
JavaScript
// Exemplo de verificação de idade
const idade = 18;

if (idade < 18) {
  console.log("Você é menor de idade.");
} else if (idade === 18) {
  console.log("Você completou 18 anos, agora é maior de idade.");
} else {
  console.log("Você é maior de idade.");
}
```



```
JavaScript
// Exemplo de verificação de idade
const idade = 18;

if (idade < 18) {
  console.log("Você é menor de idade.");
} else if (idade === 18) {
  console.log("Você completou 18 anos, agora é maior de idade.");
} else {
  console.log("Você é maior de idade.");
}
```



Exemplo 2: Classificação de Notas

```
JavaScript
// Exemplo de classificação de notas
const nota = 75;
```



```
if (nota >= 90) {  
    console.log("Sua nota é A.");  
} else if (nota >= 80) {  
    console.log("Sua nota é B.");  
} else if (nota >= 70) {  
    console.log("Sua nota é C.");  
} else if (nota >= 60) {  
    console.log("Sua nota é D.");  
} else {  
    console.log("Sua nota é F.");  
}
```

JavaScript



// Exemplo de classificação de notas

const nota = 75;

```
if (nota >= 90) {  
    console.log("Sua nota é A.");  
} else if (nota >= 80) {  
    console.log("Sua nota é B.");  
} else if (nota >= 70) {  
    console.log("Sua nota é C.");  
} else if (nota >= 60) {  
    console.log("Sua nota é D.");  
} else {  
    console.log("Sua nota é F.");  
}
```

Nesses exemplos, estamos usando as estruturas `if`, `else if` e `else` para realizar diferentes ações com base nas condições fornecidas. No primeiro exemplo, estamos verificando a idade e fornecendo mensagens diferentes para menores de idade, aqueles que acabaram de completar 18 anos e maiores de idade. No segundo exemplo, estamos atribuindo letras de classificação com base nas notas fornecidas.

Loops e Iterações

Os loops, como `for` e `while`, são usados para executar um bloco de código várias vezes. Isso é útil para percorrer listas de itens ou realizar tarefas repetitivas.

Exemplo 1: Utilizando o Loop `for`

JavaScript



// Exemplo de loop for para imprimir os números de 1 a 5

```
for (let i = 1; i <= 5; i++) {  
    console.log(i);  
}
```

JavaScript



// Exemplo de loop for para imprimir os números de 1 a 5

```
for (let i = 1; i <= 5; i++) {  
    console.log(i);  
}
```

}

Exemplo 2: Utilizando o Loop while

```
JavaScript
// Exemplo de loop while para calcular a soma dos números de 1 a 10
let numero = 1;
let soma = 0;

while (numero <= 10) {
    soma += numero;
    numero++;
}

console.log("A soma dos números de 1 a 10 é: " + soma);
```

```
JavaScript
// Exemplo de loop while para calcular a soma dos números de 1 a 10
let numero = 1;
let soma = 0;

while (numero <= 10) {
    soma += numero;
    numero++;
}

console.log("A soma dos números de 1 a 10 é: " + soma);
```

Nesses exemplos, estamos usando os loops for e while para realizar tarefas diferentes. No primeiro exemplo, usamos o loop for para imprimir os números de 1 a 5. No segundo exemplo, usamos o loop while para calcular a soma dos números de 1 a 10. Em ambos os casos, os loops são usados para repetir ações até que uma condição seja atendida.

Funções JavaScript

As funções são blocos de código reutilizáveis que podem receber entradas, processá-las e retornar um resultado. Vamos explorar como definir e invocar funções em JavaScript.

Exemplo 1: Definindo e Invocando uma Função Simples

```
JavaScript
// Exemplo de definição de uma função que saúda o usuário

function saudacao(nome) {
    console.log("Olá, " + nome + "! Bem-vindo.");
}

// Invocando a função com um nome

saudacao("Ana");
```

```
// Saída: Olá, Ana! Bem-vindo.
```

```
JavaScript
```

```
// Exemplo de definição de uma função que saúda o usuário
```

```
function saudacao(nome) {  
  console.log("Olá, " + nome + "! Bem-vindo.");  
}
```

```
// Invocando a função com um nome
```

```
saudacao("Ana");
```

```
// Saída: Olá, Ana! Bem-vindo.
```

Exemplo 2: Função para Calcular a Área de um Retângulo

```
JavaScript
```

```
// Exemplo de definição de uma função que calcula a área de um retângulo
```

```
function calcularAreaRetangulo(largura, altura) {  
  return largura * altura;  
}
```

```
// Invocando a função para calcular a área de um retângulo
```

```
const largura = 5;  
const altura = 8;  
const area = calcularAreaRetangulo(largura, altura);
```

```
console.log("A área do retângulo é: " + area);
```

```
// Saída: A área do retângulo é: 40
```

```
JavaScript
```

```
// Exemplo de definição de uma função que calcula a área de um retângulo
```

```
function calcularAreaRetangulo(largura, altura) {  
  return largura * altura;  
}
```

```
// Invocando a função para calcular a área de um retângulo
```

```
const largura = 5;  
const altura = 8;  
const area = calcularAreaRetangulo(largura, altura);
```

```
console.log("A área do retângulo é: " + area);
```

```
// Saída: A área do retângulo é: 40
```

Nesses exemplos, estamos definindo e invocando funções em JavaScript. No primeiro exemplo, definimos uma função simples que saúda o usuário com base no nome fornecido. No segundo exemplo, criamos uma função que calcula a área de um retângulo com base na largura e altura fornecidas. A função retorna o resultado, que é então impresso usando `console.log()`.



Exercícios Práticos de Lógica de Programação

Agora que estamos familiarizados com os conceitos básicos, é hora de enfrentar alguns exercícios práticos de lógica de programação em JavaScript.

Calculadora Simples

Vamos começar com algo simples. Crie uma calculadora que possa realizar operações de adição, subtração, multiplicação e divisão.

Exemplo 1: Calculadora Simples

```
JavaScript    
  
// Função para adição  
function adicao(a, b) {  
    return a + b;  
}  
  
// Função para subtração  
function subtracao(a, b) {  
    return a - b;  
}  
  
// Função para multiplicação  
function multiplicacao(a, b) {  
    return a * b;  
}  
  
// Função para divisão  
function divisao(a, b) {  
    if (b === 0) {  
        return "Não é possível dividir por zero";  
    }  
    return a / b;  
}  
  
console.log(adicao(5, 3)); // Saída: 8  
console.log(subtracao(10, 4)); // Saída: 6  
console.log(multiplicacao(3, 7)); // Saída: 21  
console.log(divisao(15, 3)); // Saída: 5  
console.log(divisao(10, 0)); // Saída: "Não é possível dividir por zero"
```

```
JavaScript
// Função para adição
function adicao(a, b) {
    return a + b;
}

// Função para subtração
function subtracao(a, b) {
    return a - b;
}

// Função para multiplicação
function multiplicacao(a, b) {
    return a * b;
}

// Função para divisão
function divisao(a, b) {
    if (b === 0) {
        return "Não é possível dividir por zero";
    }
    return a / b;
}

console.log(adicao(5, 3)); // Saída: 8
console.log(subtracao(10, 4)); // Saída: 6
console.log(multiplicacao(3, 7)); // Saída: 21
console.log(divisao(15, 3)); // Saída: 5
console.log(divisao(10, 0)); // Saída: "Não é possível dividir por zero"
```

Verificação de Número Primo

Desenvolva um programa que verifique se um número fornecido é primo ou não. Lembre-se de que um número primo é divisível apenas por 1 e por ele mesmo.

Exemplo 2: Verificação de Número Primo

```
JavaScript
// Função para verificar se um número é primo

function ehPrimo(numero) {
    if (numero <= 1) {
        return false;
    }

    for (let i = 2; i <= Math.sqrt(numero); i++) {
        if (numero % i === 0) {
```

```
        return false; }
    } return true;
}

console.log(ehPrimo(7)); // Saída: true
console.log(ehPrimo(12)); // Saída: false
console.log(ehPrimo(2)); // Saída: true
console.log(ehPrimo(1)); // Saída: false
```

JavaScript



// Função para verificar se um número é primo

```
function ehPrimo(numero) {
    if (numero <= 1) {
        return false;
    }

    for (let i = 2; i <= Math.sqrt(numero); i++) {
        if (numero % i === 0) {
            return false; }
    } return true;
}

console.log(ehPrimo(7)); // Saída: true
console.log(ehPrimo(12)); // Saída: false
console.log(ehPrimo(2)); // Saída: true
console.log(ehPrimo(1)); // Saída: false
```

2.3 Reversão de String

Crie uma função que aceite uma string como entrada e retorne a mesma string, mas em ordem reversa. Isso testará sua compreensão das iterações em strings.

Exemplo 1: Reversão de String

JavaScript



// Função para reverter uma string

```
function reverterString(str) {

    let reversed = "";
    for (let i = str.length - 1; i >= 0; i--) {
        reversed += str[i];
    }
    return reversed;
}

const frase = "Olá, mundo!";
```



```
const fraseRevertida = reverterString(frase);

console.log(fraseRevertida); // Saída: !odnum ,ál0
```

JavaScript



```
// Função para reverter uma string
```

```
function reverterString(str) {

    let reversed = "";
    for (let i = str.length - 1; i >= 0; i--) {
        reversed += str[i];
    }
    return reversed;
}
```

```
const frase = "Olá, mundo!";

const fraseRevertida = reverterString(frase);

console.log(fraseRevertida); // Saída: !odnum ,ál0
```

Exemplo 2: Reversão de String usando Métodos Integrados

JavaScript



```
// Função para reverter uma string usando métodos integrados
```

```
function reverterString(str) {
    return str.split("").reverse().join("");
}
```

```
const palavra = "javascript";
const palavraRevertida = reverterString(palavra);

console.log(palavraRevertida); // Saída: tpircsavaJ
```

JavaScript



```
// Função para reverter uma string usando métodos integrados
```

```
function reverterString(str) {
    return str.split("").reverse().join("");
}
```



```
const palavra = "javascript";
const palavraRevertida = reverterString(palavra);



console.log(palavraRevertida); // Saída: tpircsavaJ
```

2.4 Contador de Palavras



Desenvolva um programa que conte quantas palavras existem em uma frase fornecida. Isso envolve dividir a string em palavras individuais e contar sua quantidade.

Exemplo 1: Contador de Palavras

```
JavaScript    
// Função para contar palavras em uma frase  
  
function contarPalavras(frase) {  
  const palavras = frase.split(" ");  
  return palavras.length;  
}  
  
const texto = "A programação é divertida e desafiadora."  
const quantidadePalavras = contarPalavras(texto)  
  
console.log("O texto possui " + quantidadePalavras + " palavras."); // Saída: O texto poss
```

```
JavaScript    
// Função para contar palavras em uma frase  
  
function contarPalavras(frase) {  
  const palavras = frase.split(" ");  
  return palavras.length;  
}  
  
const texto = "A programação é divertida e desafiadora."  
const quantidadePalavras = contarPalavras(texto)  
  
console.log("O texto possui " + quantidadePalavras + " palavras."); // Saída: O texto poss
```

Exemplo 2: Contador de Palavras usando Expressões Regulares

```
JavaScript    
// Função para contar palavras em uma frase usando expressões regulares  
  
function contarPalavras(frase) {  
  const regex = /\w+/g;  
  const palavras = frase.match(regex);  
  return palavras ? palavras.length : 0;  
}  
  
const texto = "Expressões regulares são poderosas na manipulação de strings!"  
const quantidadePalavras = contarPalavras(texto)  
  
console.log("O texto possui " + quantidadePalavras + " palavras."); // Saída: O texto poss
```

JavaScript



```
// Função para contar palavras em uma frase usando expressões regulares

function contarPalavras(frase) {
  const regex = /\w+/g;
  const palavras = frase.match(regex);
  return palavras ? palavras.length : 0;
}

const texto = "Expressões regulares são poderosas na manipulação de strings!";
const quantidadePalavras = contarPalavras(texto);

console.log("O texto possui " + quantidadePalavras + " palavras."); // Saída: O texto possui 12 palavras.
```

Nesses exemplos, estamos resolvendo os exercícios de reversão de string e contador de palavras. No primeiro exemplo, estamos utilizando um loop for para reverter a string e, no segundo exemplo, estamos usando os métodos split, reverse e join para alcançar o mesmo resultado. Para o contador de palavras, no terceiro exemplo estamos dividindo a string em palavras e contando sua quantidade, enquanto no quarto exemplo, estamos usando expressões regulares para contar as palavras.

Aprofundando com Desafios Intermediários

Agora que você dominou os exercícios anteriores, é hora de enfrentar desafios mais intermediários.

Fatorial de um Número

Crie uma função que calcule o fatorial de um número dado. O fatorial de um número é o produto de todos os inteiros positivos menores ou iguais a esse número.

Exemplo: Fatorial de um Número

JavaScript



```
// Função para calcular o fatorial de um número

function calcularFatorial(numero) {
  if (numero === 0 || numero === 1) {
    return 1;
  } else {
    return numero * calcularFatorial(numero - 1);
  }
}

const numero = 5;
const fatorial = calcularFatorial(numero);

console.log("O fatorial de " + numero + " é: " + fatorial); // Saída: O fatorial de 5 é: 120
```

JavaScript



```
// Função para calcular o fatorial de um número
```

```
function calcularFatorial(numero) {  
  if (numero === 0 || numero === 1) {  
    return 1; }  
  else {  
    return numero * calcularFatorial(numero - 1);  
  }  
}  
  
const numero = 5;  
const fatorial = calcularFatorial(numero);  
  
console.log("O fatorial de " + numero + " é: " + fatorial); // Saída: O fatorial de 5 é: 120
```

Fibonacci em JavaScript

Desenvolva um programa que gere a série Fibonacci até um determinado termo. A série começa com 0 e 1, e cada termo subsequente é a soma dos dois anteriores.

Exemplo: Fibonacci em JavaScript

```
JavaScript  
// Função para gerar a série Fibonacci até um termo dado  
  
function gerarFibonacci(quantidadeTermos) {  
  const fibonacci = [0, 1];  
  
  for (let i = 2; i < quantidadeTermos; i++) {  
    fibonacci[i] = fibonacci[i - 1] + fibonacci[i - 2];  
  }  
  
  return fibonacci;  
}  
  
const termosDesejados = 8;  
const serieFibonacci = gerarFibonacci(termosDesejados);  
  
console.log("Série Fibonacci até o termo " + termosDesejados + ": " + serieFibonacci); //
```

```
JavaScript  
// Função para gerar a série Fibonacci até um termo dado  
  
function gerarFibonacci(quantidadeTermos) {  
  const fibonacci = [0, 1];  
  
  for (let i = 2; i < quantidadeTermos; i++) {  
    fibonacci[i] = fibonacci[i - 1] + fibonacci[i - 2];  
  }  
  
  return fibonacci;  
}
```

```
const termosDesejados = 8;
const serieFibonacci = gerarFibonacci(termosDesejados);

console.log("Série Fibonacci até o termo " + termosDesejados + ": " + serieFibonacci); //
```

Ordenação de Array

Implemente um algoritmo de ordenação para organizar os elementos de um array em ordem crescente.

Um exemplo é o algoritmo "Bubble Sort".

Exemplo: Ordenação de Array usando Bubble Sort

JavaScript



```
// Função para ordenar um array usando o algoritmo Bubble Sort
```

```
function bubbleSort(array) {
  const tamanho = array.length;

  for (let i = 0; i < tamanho - 1; i++) {
    for (let j = 0; j < tamanho - i - 1; j++) {
      if (array[j] > array[j + 1]) {
        // Trocar elementos
        const temp = array[j];
        array[j] = array[j + 1];
        array[j + 1] = temp;
      }
    }
  }

  return array;
}

const numeros = [3, 1, 8, 5, 2, 7];
const numerosOrdenados = bubbleSort(numeros);

console.log("Array ordenado: " + numerosOrdenados); // Saída: Array ordenado: 1,2,3,5,7,8
```

JavaScript



```
// Função para ordenar um array usando o algoritmo Bubble Sort
```

```
function bubbleSort(array) {
  const tamanho = array.length;

  for (let i = 0; i < tamanho - 1; i++) {
    for (let j = 0; j < tamanho - i - 1; j++) {
      if (array[j] > array[j + 1]) {
        // Trocar elementos
        const temp = array[j];
        array[j] = array[j + 1];
        array[j + 1] = temp;
      }
    }
  }

  return array;
}
```

```
        array[j + 1] = temp;
    }
}

return array;
}

const numeros = [3, 1, 8, 5, 2, 7];
const numerosOrdenados = bubbleSort(numeros);

console.log("Array ordenado: " + numerosOrdenados); // Saída: Array ordenado: 1,2,3,5,7,8
```

Nesses exemplos, estamos resolvendo os exercícios de fatorial de um número, série Fibonacci e ordenação de array. No primeiro exemplo, estamos calculando o fatorial de um número usando uma função recursiva. No segundo exemplo, estamos gerando a série Fibonacci até um termo desejado. No terceiro exemplo, estamos implementando o algoritmo "Bubble Sort" para ordenar os elementos de um array em ordem crescente.

Aplicando Lógica à Programação JavaScript Avançada

Agora que estamos nos aproximando do nível avançado, é hora de aplicar nossa lógica a desafios mais complexos.

Manipulando Objetos

Crie um programa que demonstre como criar e manipular objetos em JavaScript. Isso inclui adicionar propriedades, acessar valores e realizar iterações.

Exemplo: Manipulando Objetos em JavaScript

```
JavaScript

// Criando um objeto representando uma pessoa
const pessoa = { nome: "João", idade: 30, profissao: "Engenheiro", };

// Acessando valores do objeto
console.log("Nome: " + pessoa.nome); // Saída: Nome: João
console.log("Idade: " + pessoa.idade); // Saída: Idade: 30
console.log("Profissão: " + pessoa.profissao); // Saída: Profissão: Engenheiro

// Adicionando uma nova propriedade ao objeto
pessoa.cidade = "São Paulo";

// Acessando a nova propriedade
console.log("Cidade: " + pessoa.cidade); // Saída: Cidade: São Paulo

// Iterando sobre as propriedades do objeto
for (let chave in pessoa) {
    console.log(chave + ": " + pessoa[chave]);
}
```



```
/* Saída:
Nome: João
Idade: 30
Profissão: Engenheiro
Cidade: São Paulo
*/

// Removendo uma propriedade do objeto
delete pessoa.profissao;

// Verificando se uma propriedade existe no objeto
if ("profissao" in pessoa) {
    console.log("A pessoa ainda possui uma profissão.");
} else {
    console.log("A pessoa não possui mais uma profissão.");
}
```

JavaScript



```
// Criando um objeto representando uma pessoa
const pessoa = { nome: "João", idade: 30, profissao: "Engenheiro", };

// Acessando valores do objeto
console.log("Nome: " + pessoa.nome); // Saída: Nome: João
console.log("Idade: " + pessoa.idade); // Saída: Idade: 30
console.log("Profissão: " + pessoa.profissao); // Saída: Profissão: Engenheiro

// Adicionando uma nova propriedade ao objeto
pessoa.cidade = "São Paulo";

// Acessando a nova propriedade
console.log("Cidade: " + pessoa.cidade); // Saída: Cidade: São Paulo

// Iterando sobre as propriedades do objeto
for (let chave in pessoa) {
    console.log(chave + ": " + pessoa[chave]);
}

/* Saída:
Nome: João
Idade: 30
Profissão: Engenheiro
Cidade: São Paulo
*/

// Removendo uma propriedade do objeto
delete pessoa.profissao;

// Verificando se uma propriedade existe no objeto
if ("profissao" in pessoa) {
    console.log("A pessoa ainda possui uma profissão.");
}
```

```
} else {  
  console.log("A pessoa não possui mais uma profissão.");  
}
```

Neste exemplo, estamos demonstrando como criar e manipular objetos em JavaScript. Começamos criando um objeto chamado `pessoa` com algumas propriedades. Em seguida, mostramos como acessar os valores das propriedades, adicionar uma nova propriedade, iterar sobre as propriedades usando um loop `for...in`, remover uma propriedade e verificar se uma propriedade existe no objeto. Isso nos dá uma visão geral de como manipular objetos em JavaScript.

Conclusão

Parabéns por percorrer esta jornada de exercícios de lógica de programação em JavaScript! Esperamos que esses desafios tenham ajudado a fortalecer suas habilidades algorítmicas e de resolução de problemas. Lembre-se de que a prática constante é fundamental para o aprimoramento contínuo.

Agora é hora de continuar praticando e explorando projetos mais complexos. Se você quiser elevar ainda mais suas habilidades, não deixe de experimentar problemas do mundo real e colaborar com outros programadores.

Perguntas frequentes

1. Posso fazer esses exercícios em qualquer ordem?

Sim, você pode escolher a ordem que melhor se adapta ao seu nível de habilidade e interesse.

2. Preciso ter experiência prévia em programação?

Não é necessário. Esses exercícios são projetados para todos, desde iniciantes até programadores experientes.

3. O que devo fazer se eu ficar preso em um exercício?

Tente dividir o problema em etapas menores e resolver cada etapa individualmente. Se ainda estiver com dificuldades, procure recursos online ou peça ajuda em comunidades de programação.

4. Posso usar recursos externos ao resolver esses exercícios?

Claro! Usar recursos externos, como documentações e tutoriais, faz parte do processo de aprendizado e resolução de problemas.

5. Onde posso encontrar projetos reais para aplicar minhas habilidades?

Plataformas de desenvolvimento, como GitHub e GitLab, frequentemente têm projetos de código aberto que você pode contribuir. Isso proporciona uma experiência valiosa no mundo real.

- Variáveis JavaScript: Guia Completo para Iniciantes
- Javascript
- Bancos de Dados : Introdução
- SGBDs – Sistemas Gerenciadores de Banco de Dados: O que é e como funcionam?
- SQL o que é? Guia Explicativo e Conciso





Prof. Eduardo H Gomes

Mestre em Engenharia da Informação, Especialista em Engenharia da Computação, Cientista da Computação, Professor de Inteligência Artificial no IFSP, 18 anos de docência no Ensino Superior. Apaixonado por Surf, Paraglider, Mergulho livre, Tecnologia, SEO, Banco de Dados e Desenvolvimento Web.



	<h3>Matemática para Idades de 5-12</h3> <p>Mais de 800 tópicos de matemática, para 5 e 12 anos. Jogos, atividades e muito mais.</p>
IXL	Abrir

Matemática para Idades de 5-12

Mais de 800 tópicos de matemática, para 5 e 12 anos. Jogos, atividades e muito mais.

IXL

A

Tipos de Dados em JavaScript o que é: Descubra agora!

Operador Ternário JavaScript: Uma Alternativa Concisa ao if else

Exercícios de Lógica de Programação em JavaScript: Aprimore suas Habilidades

Como fazer jejum intermitente

Emagreça com dieta cetogênica, dieta vegana ou jejum intermitente acordo com sua idade.

FastEasy

A

Fique à frente do jogo tecnológico!

Inscreva-se na nossa newsletter agora.

Inscreva-se



Tech News

Reviews
Software
Linux
Python
Segurança
Criptografia
Google
Excel
Games
Informática

Quem Somos

Elearning
Ensino
Sobre
Contato

Legal

Política de Privacidade
Termos e condições
Política de Cookies
Mapa do Site

