

Curso de JavaScript

1. Executando JavaScript no navegador

Primeiro, dentro da pasta Explorador de Arquivos, criamos uma pasta que daremos o nome de “projetos” e a abrimos usando o visual studio code. Do mesmo modo como foi apresentado no curso de html e css, dentro da pasta PROJETOS aberta no *vscode*, criamos um arquivo sob o nome de *index.html* e nele geramos a estrutura básica de um documento html, digitando ! e teclando Enter. Vamos começar digitando na área corpo do html as tags **<script>** e **</script>**. Em seguida, dentro da tag script, digitamos a função *console.log*, que imprime um texto no navegador.



```
> index.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Introdução</title>
7  </head>
8  <body>
9      <script>
10         console.log("Olá Mundo");
11     </script>
12 </body>
13 </html>
```

Figura 1.

Vamos então abrir o arquivo *index.html* no Google Chrome. Abre-se então uma tela em nosso navegador sem nenhum documento escrito. Porém, clicando com o botão direito do mouse sobre qualquer área da tela, abre-se uma pequena janela de opções e ali selecionamos Inspecionar. Acessamos então o DevTools e em um de seus menus, ao lado da aba Elements, existe a aba Console. Ao clicar sobre esta aba, é aberto um espaço na tela onde podemos observar o parâmetro da função *console.log* sendo executado, o que demonstra a conexão do JavaScript com o navegador.

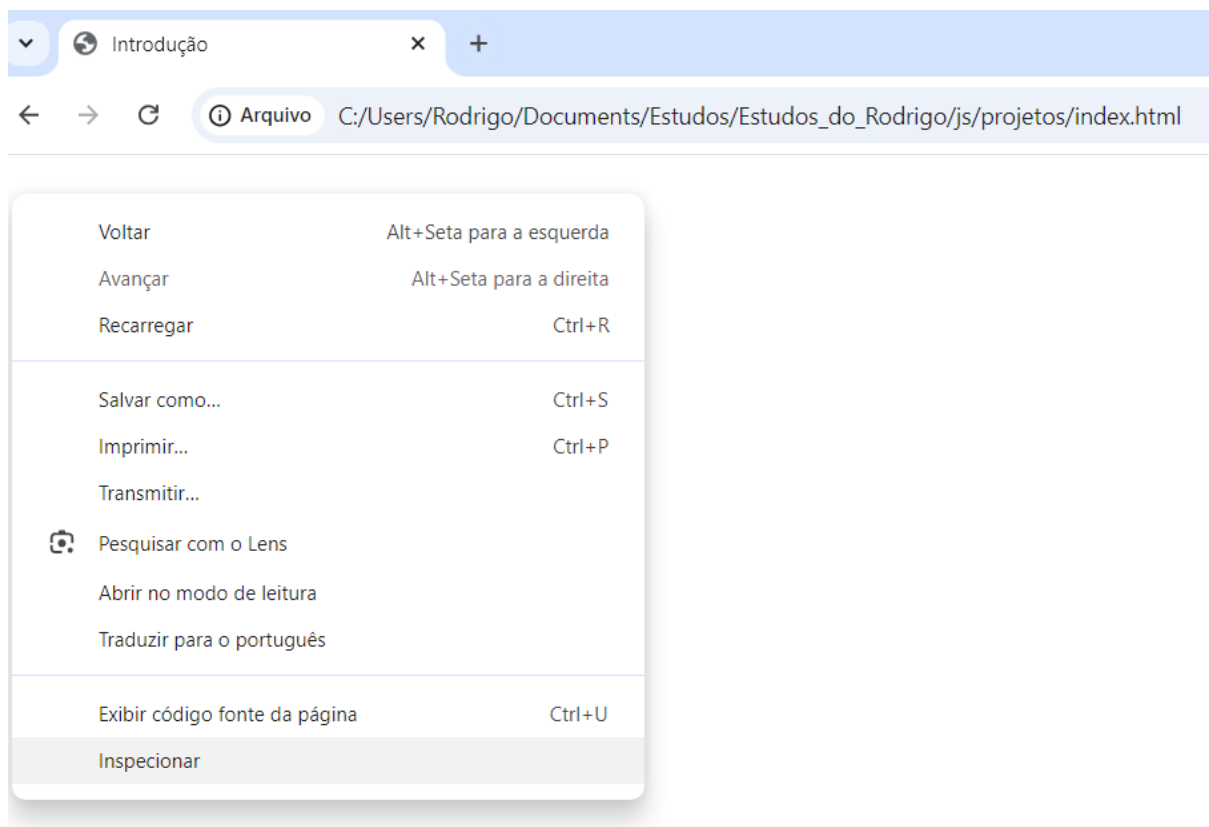


Figura 2.

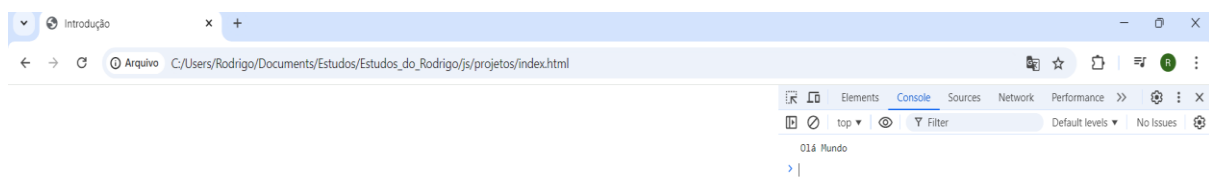
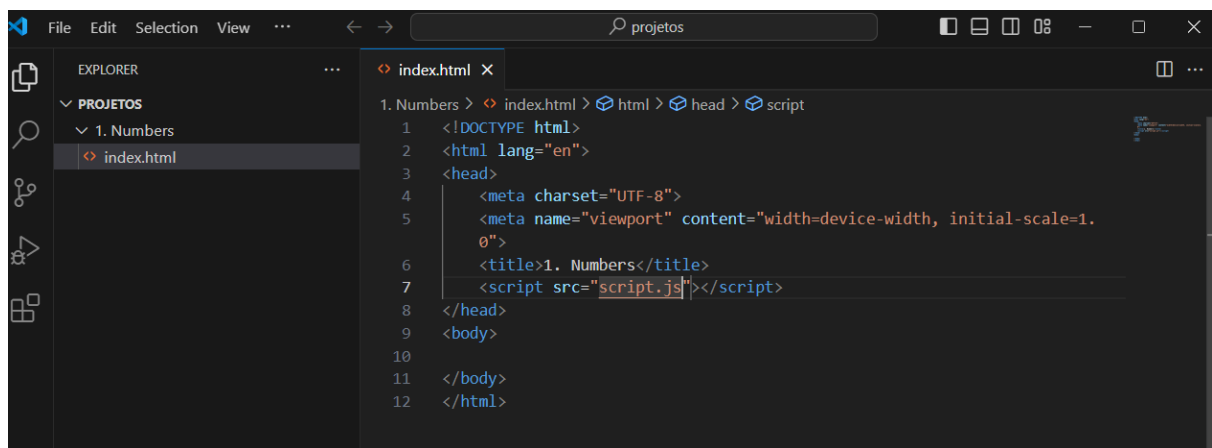


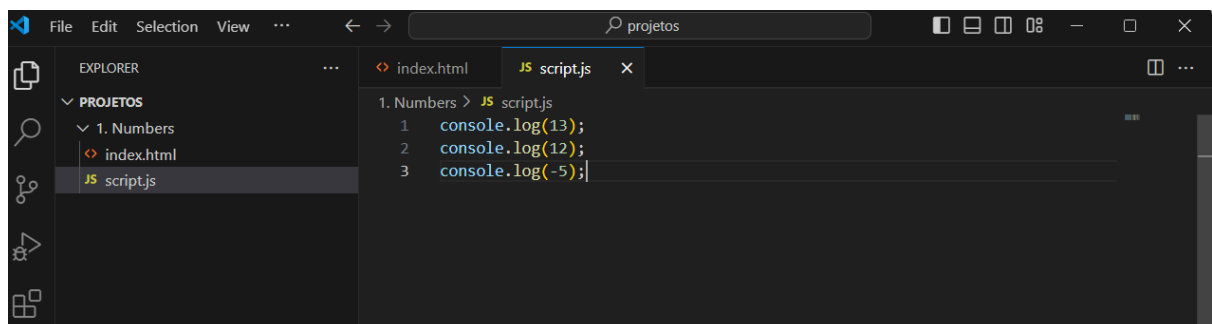
Figura 3.

2. Números em JavaScript

Vamos abrir a nossa pasta projetos com o Code e logo abaixo do nome PROJETOS no VS Code vamos clicar com o botão direito do mouse e selecionar a opção New Folder e criar uma nova pasta chamada **1. Numbers**. Em seguida, clicamos com o botão direito do mouse sobre o nome da pasta criada e na janela de opções que aparecerá, selecionamos New File. No campo que aparecerá para nomear o arquivo escrevemos index.html, clicamos em Enter e no documento gerado, criamos a estrutura de documento html. Na área head, logo abaixo de title, inserimos a tag script. A figura abaixo exemplifica este processo.



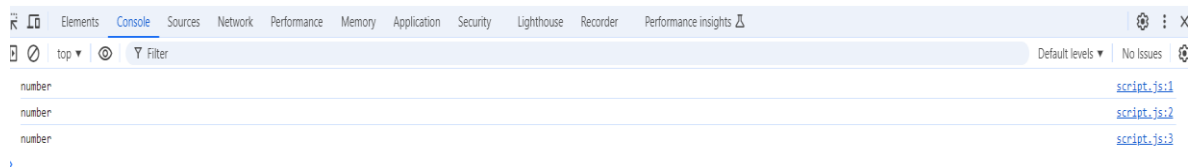
Em seguida, criamos um novo arquivo sob o nome script.js e no documento gerado digitamos a função console.log e inserimos como parâmetro alguns números.



Abrimos então o arquivo index.html no Google Chrome e clicamos com o botão direito do mouse sobre qualquer área da tela aberta. Na janela de

opções que aparecerá em seguida, selecionamos Inspeccionar para acessarmos o DevTools. Nele, selecionamos o menu console, que nos apresentará os resultados impressos. A função `typeof` da maneira como segue na figura abaixo revela no navegador a real natureza desses parâmetros em JS: todos são *numbers*.

```
<> index.html JS script.js X  
1. Numbers > JS script.js  
1 console.log(typeof 13);  
2 console.log(typeof 12.7);  
3 console.log(typeof -5);
```

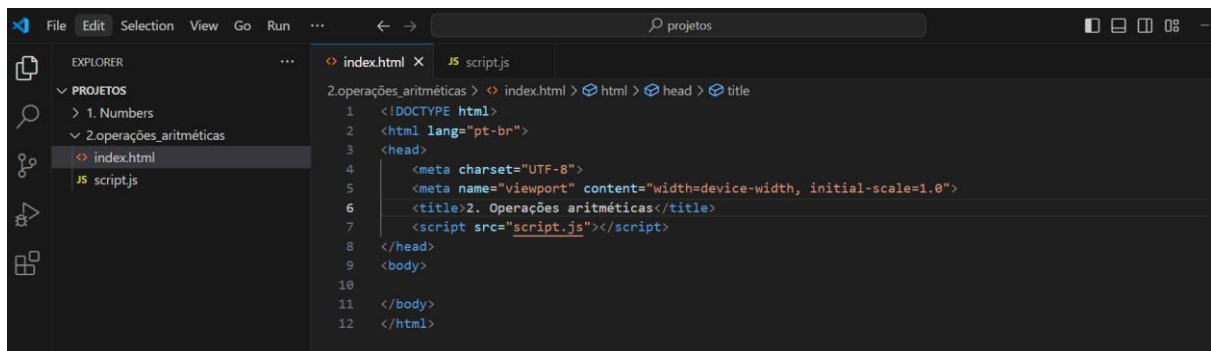


3. Operações aritméticas

Em JavaScript, as operações básicas da aritmética são realizadas com uso dos seguintes operadores.

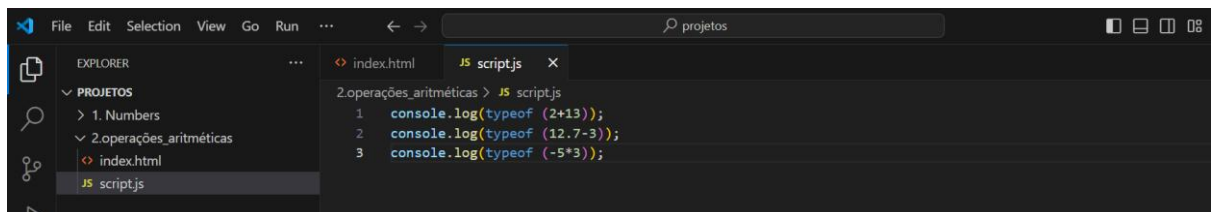
+	Adição
-	Subtração
*	Multiplicação
/	Divisão
%	Resto

Em nossa pasta projetos criamos uma nova pasta com o título 2. operações_aritméticas e nela criamos um arquivo index.html e um arquivo script.js. As figuras abaixo trazem uma representação desta situação.



The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows a project named 'PROJETOS' with a folder '2.operações_aritméticas' containing 'index.html' and 'script.js'. The main editor area shows the 'index.html' file with the following code:

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>2. Operações aritméticas</title>
7   <script src="script.js"></script>
8 </head>
9 <body>
10
11 </body>
12 </html>
```



The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows the same project structure. The main editor area shows the 'script.js' file with the following code:

```
1 console.log(typeof (2+13));
2 console.log(typeof (12.7-3));
3 console.log(typeof (-5*3));
```

4. Divisibilidade

Se a e b são inteiros, dizemos que a divide b , denotamos $a|b$, quando existe um inteiro c tal que $b = ac$. Se a não divide b , escrevemos $a \nmid b$.

Podemos montar um programa em JavaScript que nos mostra se $a|b$ ou não e, em caso afirmativo, imprime na tela o resultado dessa divisão.

Primeiro, criamos o nosso código html.



```
<? index.html X
<? index.html > ...
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Divisibilidade</title>
7      <script id="MathJax-script" async src="https://cdn.jsdelivr.net/npm/mathjax@3/es5/tex-mml-
      js"></script>
8      <script src="script.js"></script>
9  </head>
10 <body>
11   <h1>Divisibilidade</h1>
12   <p>Se  $(a)$  e  $(b)$  são inteiros, dizemos que  $(a)$  divide  $(b)$  se existir um inteiro  $(c)$  tal que
       $(b=a\cdot c)$  </p>
13
14 </body>
15 </html>
```

Em nosso html acima estamos utilizando um mecanismo de exibição de texto matemático, o MathJax. Para tanto, inserimos no código html o código seguinte

```
<script id="MathJax-script" async
      src="https://cdn.jsdelivr.net/npm/mathjax@3/es5/tex-mml-
      chtml.js">
```

O código acima é encontrado através do endereço abaixo.

<https://www.mathjax.org/#gettingstarted>

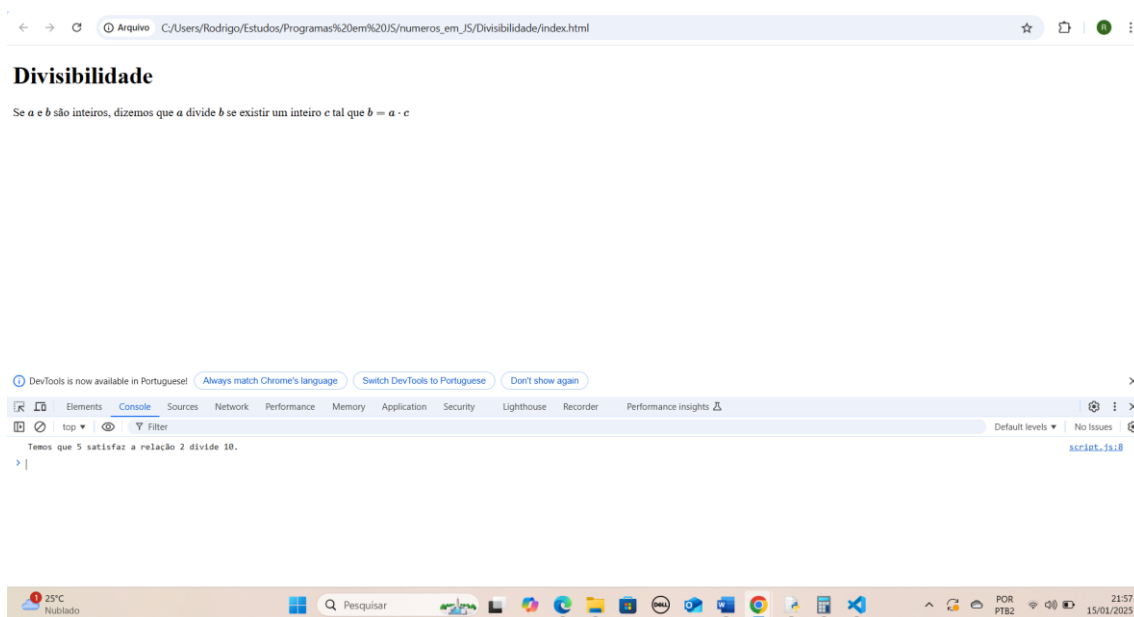
Com este mecanismo, para reproduzir um texto matemático em nossa página html, podemos utilizar a notação em LaTeX e delimitar o texto com o delimitador $(...)$, tal como se vê no código acima.

Após criar o nosso código html, criamos então o nosso arquivo script.js para salvar o nosso código em JavaScript. Em JS, declaramos uma variável cujo valor não é constante utilizando a palavra-chave **let**.

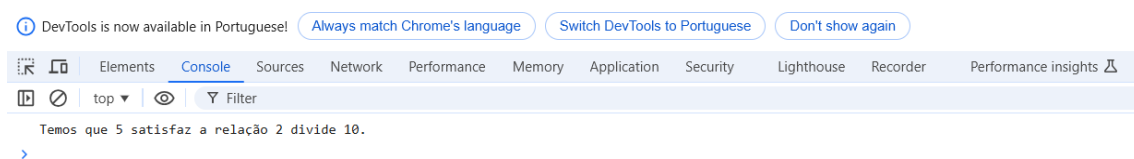
```
<let nome_da_variável> = <valor_atribuído_a_variável>
```

```
JS script.js M X
JS script.js > ...
1 let b = prompt('Digite um número inteiro para ser o dividendo');
2 let a = prompt('Digite um segundo número inteiro para ser o divisor');
3 let c = b/a;
4 let d = b%a; // Esta operação resulta no valor do resto da divisão de b por a.
5 if(d > 0) {
6     console.log(`Não existe um inteiro n tal que n multiplicado por ${a} seja igual a ${b}`);
7 } else {
8     console.log(`Temos que ${c} satisfaz a relação ${a} divide ${b}.`);
9 }
10
```

Inspecionando o nosso código, tendo como entradas $b = 10$ e $a = 5$, obtemos



No console do DevTools



Proposição 1. *Dados a, b e c inteiros, se $a|b$ e $b|c$, então $a|c$.*

É fácil ver que o enunciado da proposição acima é verdadeiro. Já a demonstração da próxima proposição pode não ser tão trivial.

Proposição 2. *Dados $a, b, c, m, n \in \mathbb{Z}$, se $c|a$ e $c|b$ então $c|(ma + nb)$.*

Demonstração: Como $c|a$ e $c|b$, $\exists r, s \in \mathbb{Z}$ tais que $a = rc$ e $b = sc$. Multiplicando um inteiro m qualquer por ambos os membros da igualdade $a = rc$, obtemos $ma = mrc$. De modo análogo, obtemos $nb = nsc$.

Somando ma e nb , obtemos $ma + nb = (mr + ns)c$, donde vem que $c|ma + nb$.

■

Pelo mesmo motivo da proposição 1, a próxima proposição também não será demonstrada.

Proposição 3. *A divisão tem as seguintes propriedades:*

- (i) $n|n$
- (ii) $d|n \Rightarrow ad|an$
- (iii) $ad|an$ e $a \neq 0 \Rightarrow d|n$
- (iv) $1|n$
- (v) $n|0$
- (vi) $d|n$ e $n \neq 0 \Rightarrow |d| \leq |n|$
- (vii) $d|n$ e $n|d \Rightarrow |d| = |n|$
- (viii) $d|n$ e $d \neq 0 \Rightarrow \left(\frac{n}{d}\right)|n$.

Até o momento estamos considerando o processo de divisão exata, que é este quando ocorre que um inteiro b pode ser tomado como o produto de dois fatores a e c . Vamos ampliar o conceito de divisão para incluir os casos em que a divisão não é exata.

Teorema 1. (Teorema da Divisão) *Dados dois inteiros a e b , $b > 0$, existe um único par de inteiros q e r tais que*

$$a = qb + r, \text{ com } 0 \leq r < b.$$

Demonstração: Se $a < b$, temos então que $\exists r \in \mathbb{Z}$ tal que $b = a + r \Leftrightarrow a = b - r$. Logo podemos tomar $q = 1$ e $r \in \mathbb{Z}_{<0}$. Suponhamos $a \geq b > 0$, vamos considerar o conjunto $X = \{x \in \mathbb{N}; bx < a\}$. Temos $X \neq \emptyset$, pois $1 \in X$. No entanto, não podemos ter $X = \mathbb{N}$, pois se assim fosse, teríamos um inteiro a maior que todo natural x , o que é um absurdo. Logo $\exists q \in X$ tal que $q > x, \forall x \in X$. Segue-se que, dado $q' = q + 1$, temos $q' \in \mathbb{N} - X$, o que implica $a \leq bq'$. De fato, se fosse $bq' < a$, $q' \in X$, o que contradiz a afirmação de que q é o maior elemento de X . Concluimos, portanto, que $\exists q \in \mathbb{Z}$ tal que $bq < a \leq bq'$. Segue-se da desigualdade $bq < a$ que $\exists r \in \mathbb{Z}$ tal que $a = bq + r$.

Suponhamos que existem dois pares $(q, r) \in \mathbb{Z} \times \mathbb{Z}$ e $(q', r') \in \mathbb{Z} \times \mathbb{Z}$ tais que $a = q'b + r' = qb + r$, com $0 \leq r' < b$ e $0 \leq r < b$. Assim,

$$\begin{cases} 0 \leq r' < b \\ 0 \leq r < b \end{cases} \Rightarrow r' - r < b \Rightarrow 0 \leq (q - q')b < b \Rightarrow 0 \leq q - q' < 1.$$

Logo temos $q - q' \in \mathbb{Z}$ e $0 \leq q - q' < 1$, o que é um absurdo. Portanto, o par $(q, r) \in \mathbb{Z} \times \mathbb{Z}$ tal que $a = qb + r$ é único.

■

O teorema da divisão serve de base para o importante algoritmo da divisão. No livro *Números Inteiros e Criptografia RSA*, de S. C. Coutinho, obtemos a seguinte descrição deste algoritmo.

Algoritmo da divisão

Entrada: inteiros positivos a e b .

Saída: inteiros não-negativos q e r tais que $a = bq + r$ e $0 \leq r < b$.

Etapa 1: Comece fazendo $Q = 0$ e $R = a$.

Etapa 2: Se $R < b$ escreva o quociente é Q e o resto é R e pare; senão vá para a Etapa 3.

Etapa 3: Se $R \geq b$ subtraia b de R e incremente Q de 1 e volte à Etapa 2.

Podemos montar o código em JS para o algoritmo da divisão. Primeiro criamos a pasta ALGORITMO_DE_DIVISAO e nela criamos os arquivos index.html e script.js com os respectivos códigos descritos abaixo.

```
index.html > html
1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <link rel="shortcut icon" href="imagens/favicon_io/favicon.ico" type="image/x-icon">
7   <title>Divisibilidade</title>
8   <script id="MathJax-script" async src="https://cdn.jsdelivr.net/npm/mathjax@3/es5/tex-mml-ctml.js"></script>
9   <script src="script.js"></script>
10 </head>
11 <body>
12   <h1>Algoritmo de divisão</h1>
13   <p>O algoritmo de divisão tem como fundamento o <em>teorema de divisão</em> que nos diz o seguinte:</p>
14   <p><em>Dados dois inteiros \(\a\) e \(\b\), \(\b \neq 0\), existe um único par de inteiros \(\q\) e \(\r\) tais que</em></p>
15   <p><math>a = bq + r</math> com <math>0 \leq r < b</math></p>
16
17 </body>
18 </html>
```

```
JS script.js > ...
1  let a = prompt('Digite um número inteiro para ser o dividendo');
2  let b = prompt('Digite um segundo número inteiro para ser o divisor');
3  let Q = 0;
4  let r = a-b*Q;
5
6  while(r>=b){
7      r = r - b;
8      Q = Q + 1;
9  }
10
11  console.log(`O quociente é ${Q} e o resto é ${r}`);
12
13  // Testando o programa para, por exemplo, a = 25 e b = 3, temos que quando for Q = 7 e r = 4, como 4 > 3,
14  // o programa realizará suas últimas operações: r = 4 - 3 e Q = 7 + 1.
15  // Obtemos então o resultado esperado, Q = 8 e r = 1.
16
```