**JAVA SCRIPT UDEMY**

**Seção 02**

**AULA 10**

* Vimos o comando console.log(‘mensagem’)

**AULA 11**

* Comentários são textos ignorados pelo editor de texto, para fazer anotações nos códigos ou fazer com que eles sejam ignorados
* Podemos usar utilizando os símbolos “//” para comentários em uma linha e “/\* \*/” para comentários maiores

**AULA 12**

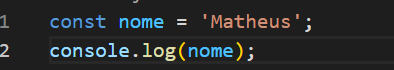
* Criamos dois arquivos, um HTML e um outro JavaScript
* Utilizamos o comando Window.alert(‘mensagem’) para exibirmos uma mensagem ao ser aberta no navegador

**AULA 13**

* Criamos nossa primeira variável utilizando o seguindo comando:
  + **Log nome = ‘João”;**
* Utilizamos ele junto do comando console.log
* Não podemos criar variáveis com palavras reservadas pela linguagem
* Devem ter nomes significantes
* Não podem começar com números
* Não podem conter espaços ou traços
* Para nomes compostos, utilizamos a camelCase
* São case sensitive
* É recomendado utilizar o Let ao invés do Var

**AULA 14**

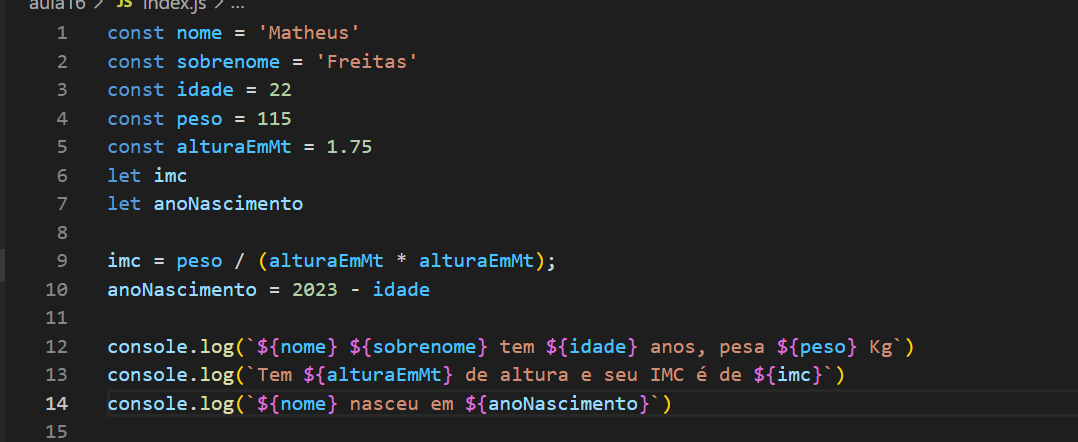
* Constantes são variáveis que não muda o seu valor
* Não podemos criar variáveis com palavras reservadas pela linguagem
* Devem ter nomes significantes
* Não podem começar com números
* Não podem conter espaços ou traços
* Para nomes compostos, utilizamos a camelCase
* São case sensitive
* É recomendado utilizar o Let ao invés do Var
* A constante precisa ser inicializada no momento que é declarado



* Realizamos algumas operações utilizando const
* Vimos o comando “typeof” usado para descobrir o tipo de dado de um elemento

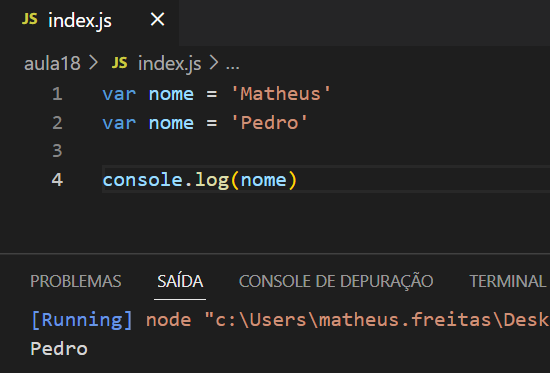
**AULA 16**

* Fizemos um exercício declarando variável utilizando const e let
* Utilizamos template Strings



AULA 18

* Nesta aula, vimos a diferença entre let e var
* Com var, é possível redeclarar a variável, diferente de let. Exemplo:

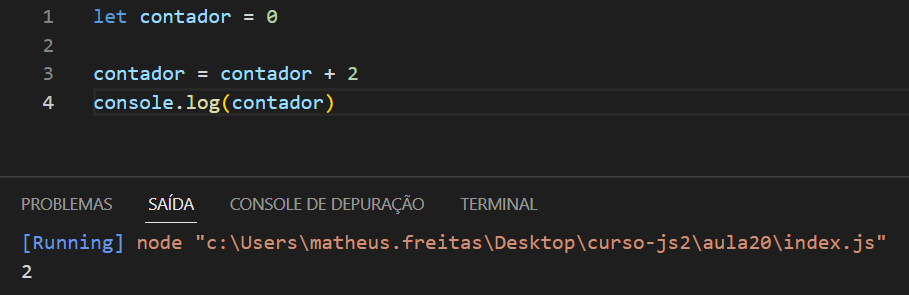


**AULA 19**

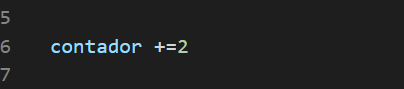
* Vimos os tipos primitivos de dados no Javascript
  + String - Texto
  + Number - Qualquer valor numérico
  + Null – Não aponta para local nenhuma na memória
  + Boolean – Verdadeiro ou Falso
* Utilizamos o comando typeof para verificar os diferentes tipos de dados

**AULA 20**

* Vimos operadores aritméticos, que são:
  + + (adição e concatenação)
  + – (subtração)
  + / (divisão)
  + \* (multiplicação)
  + \*\* (potência)
  + % (resto da divisão inteira)
* Existe uma ordem de prioridade na execução das operações
* O operador de incremento (++) adiciona o valor 1 a uma variável (não podendo ser utilizado com uma const)
* O operador de incremento utiliza a mesma lógica acima (--)
* Para fazer o incremento com um valor maior do que 1, podemos escolher o valor a ser incrementado, exemplo:



* Podemos até mesmo utilizar uma variável para identificar o valor a ser incrementado
* Podemos representar o comando acima da seguinte forma:



* Podemos converter um dado String para Number utilizando o comando ParseInt, ParseFloat ou Number

**AULA 21**

* Vimos os comandos utilizando window
  + .alert
  + .confirm
  + .prompt
* Podemos utilizar o retorno de prompt e confirm como variável
* O retorno do prompt sempre será uma string, podemos utilizar o comando Number para conseguir um retorno numérico

**AULA 24**

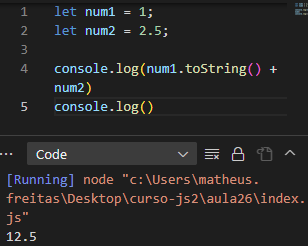
* Aulas sobre String
* Para iniciar uma variável do tipo String, temos que iniciar ela dentro de aspas simples ou aspas
* Para colocar aspas dentro do texto de uma string, podemos utilizar contra barra. Exemplo:
  + Let texto = “Um \”texto\””;
* Strings são indexadas – Cada caractere possui um número índice
* É possível selecionarmos apenas um elemento de uma String utilizando seu índice usando colchetes ou o comando Charat(8)
* O JavaScript tem um menu com elementos que permitem manipular uma String, como o comando concat (usado para concatenar elementos strings) ou CharAt (para seleciona um indice de uma String)

**EX03**

* Para escrever no arquivo HTML, podemos utilizar o comando document.write, porém o comando mais utilizado é o
  + Window.document.body.innerHTML = ‘Texto’

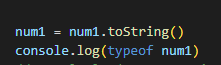
**AULA 26**

* Podemos transformar um dado number em string temporariamente utilizando o comando toString(). Exemplo:

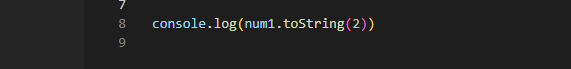


Neste exemplo, por num1 se tornar uma String, não há uma soma entre os números e sim uma concatenação dos elementos

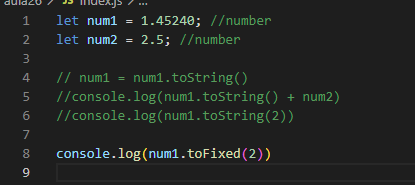
* Podemos também transformar um number de vez em String utilizando o comando toString da seguinte forma:



* Conseguimos visualizar a representação binária de um número também utilizando a função toString. Exemplo:



* É possível também representar um número com grandes casas decimais utilizando o comando toFixed. Exemplo:



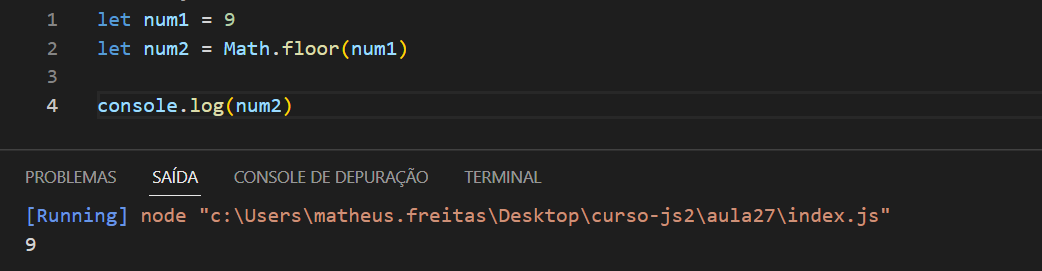
* Podemos também descobrir se um número é inteiro utilizando o seguinte comando:



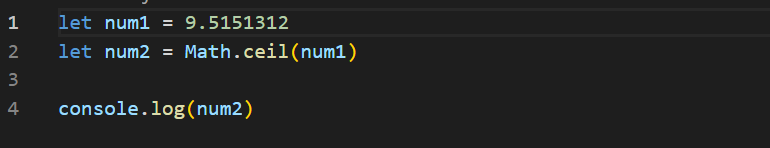
* No Js, os números são imprecisos

**AULA 27**

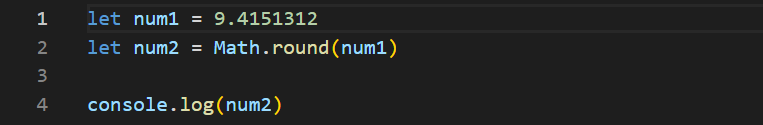
* Aula sobre o objeto Math
* Com o comando math.floor() podemos arredondar valores com casas decimais para o valor menor. Exemplo:



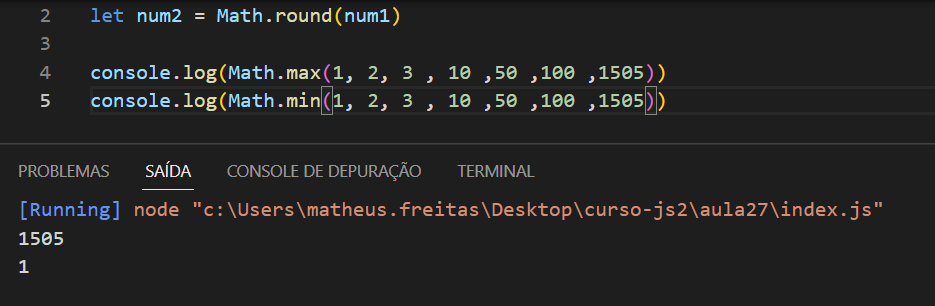
* Já com o comando math.ceil() conseguimos arredondar o valor para um valor maior. Exemplo:



* Math.roud() é uma função que utilizado para arredondar para o número inteiro mais próximo, tendo o mesmo efeito que os comandos acima.



* Existe ainda os comandos math.max e math.min, utilizados para descobrir o maior ou menor número de uma sequência numérica. Exemplo:



* Já o comando math.random() é usado para criar números aleatórios que possuem casas decimais entre 0 e 1

**AULA 28**

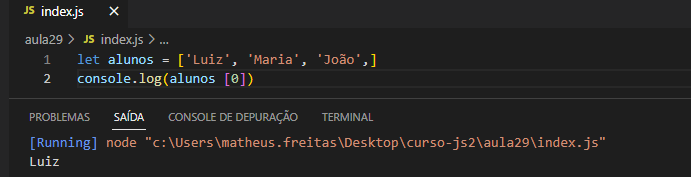
* Realizamos um exercício utilizando o innerHTML
* Criamos um arquivo HTML, utilizando as tags H1 e <div></div>
* Utilizamos a tag id para criarmos seletores para o Javascript, permitindo manipularmos desta forma
* Transformamos o valor do prompt em Number
* Utilizamos os métodos Number e Math para manipular os números

**AULA 29**

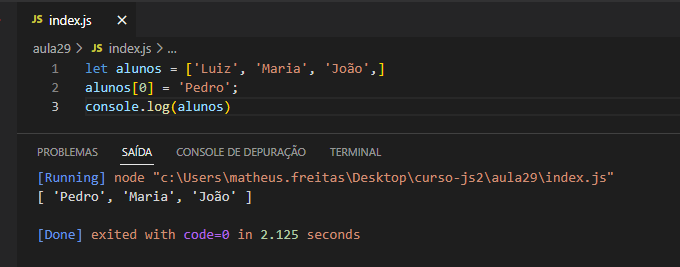
* Aula sobre arrays
* Para criarmos as arrays, é semelhante a criação de uma variável comum, basta inserir os valores dentros de colchetes. Exemplo:



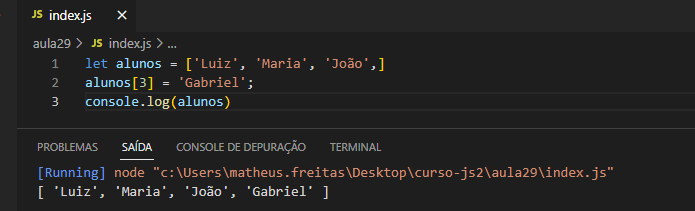
* É uma boa pratica criar arrays utilizando somente um tipo de dado, sendo string ou number por exemplo
* Possuem índices onde cada elemento (no exemplo acima, Luiz, por exemplo receberia o índice 0, Maria o índice 1 e João o índice 2)
* Para visualizarmos apenas um elemento de uma array, utilizamos o índice para a identificação. Exemplo:



* Podemos editar um valor de uma array, utilizando o mesmo método acima. Exemplo:



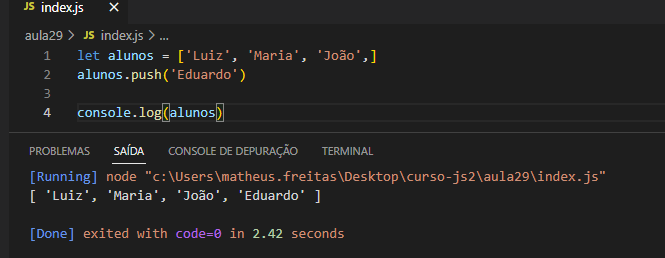
* Podemos também adicionar novos valores nas Arrays, como no exemplo acima, porém escolhendo o um índice que não foi preenchido ainda. Exemplo



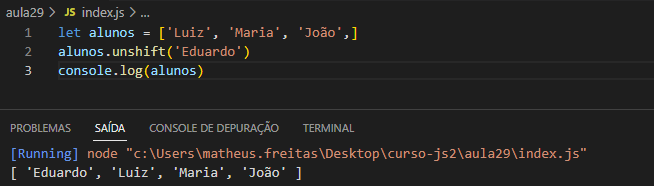
* Conseguimos descobrir o tamanho de uma array utilizando o comando .lenth



* Existe um comando que é utilizado para adicionar um elemento dentro da array. O comando é o push. Exemplo:



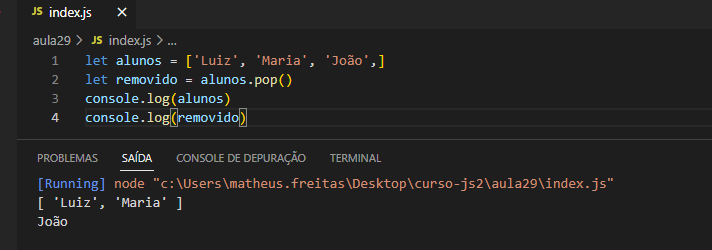
* Para adicionar um elemento no primeiro índice da array, utilizamos o comando . unshift(‘elemento’)



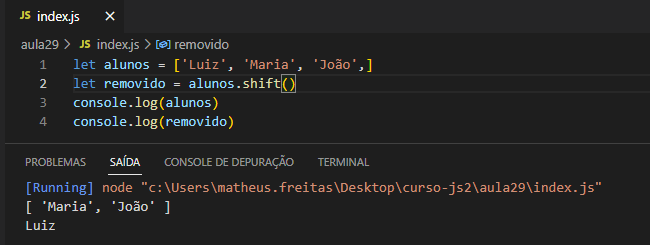
* Este comando é usado para remover o ultimo índice de uma variável



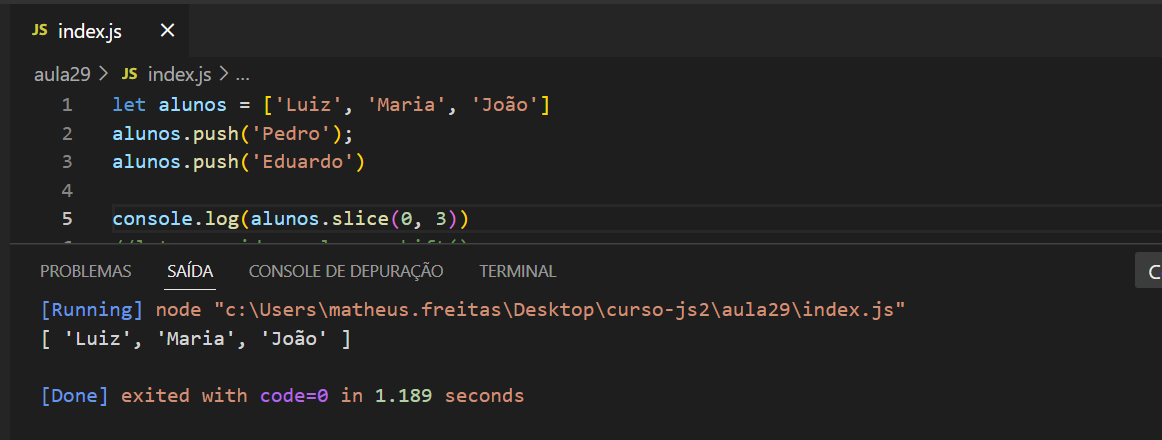
* Podemos atribuir o elemento removido de uma array em uma variável, exemplo



* Para remover o primeiro elemento, utilizamos o comando .shift()



* Podemos excluir um elemento utilizando o comando delete nomedavariavel[índice]
* Para exibirmos uma determinada faixa de elemento, utilizamos o comando slice. Exemplo:

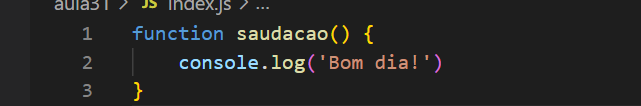


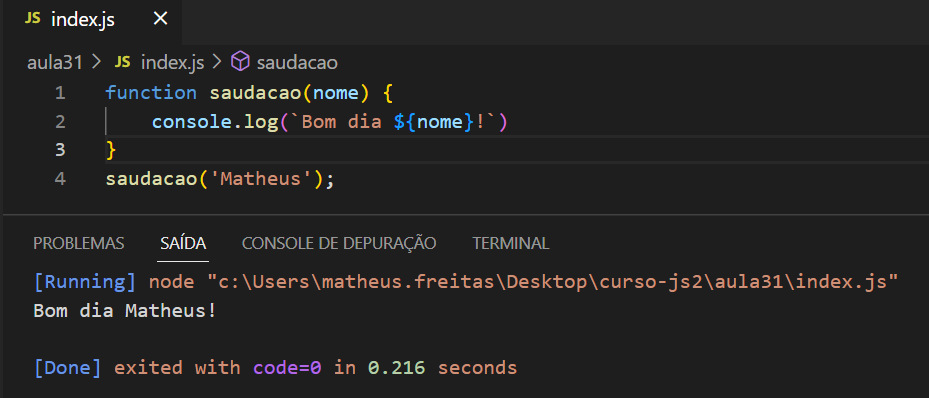
* Podemos utilizar o comando .slice com números negativos
* Uma array é considero um elemento objeto
* Para descobrirmos se uma variável é um array, podemos utilizar o comando instaceof Array

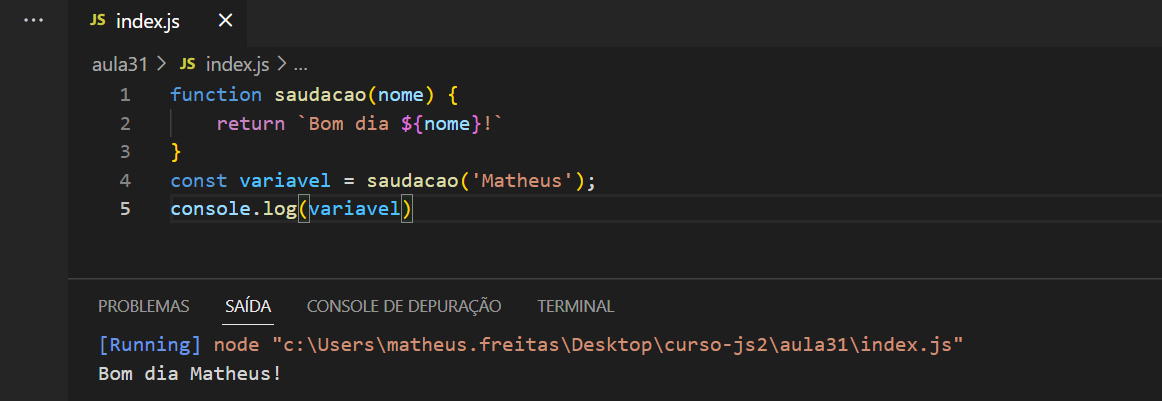


**AULA 31**

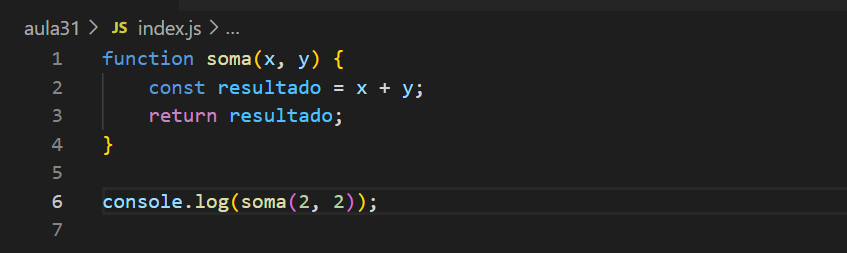
* Aula sobre funções
* Trecho de código que executa alguma ação
* Geralmente retorna um valor
* Utilizamos o comando function nomedafuncap() {}
* Uma estrutura básica de uma função é composta da seguinte maneira:



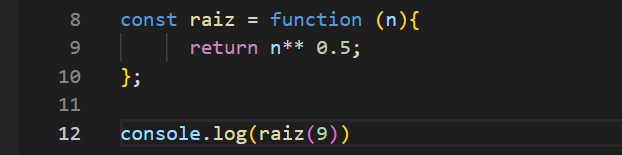
* Dentro de funções, podemos utilizar parâmetros (funcionam de forma parecia a variável)
* Com os parâmetros, podemos armazenar dados temporariamente dentro de uma função
* Para utilizar o parâmetro, basta inserir ele dentro dos colchetes após o nome da variável. Exemplo: 
* Para que a função te retorne um valor, utilizamos o comando return. Exemplo:



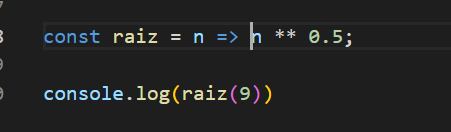
* Podemos até realizar cálculos utilizando parâmetros. Exemplo:



* Após o comando return, nada mais será executado
* No exemplo da soma, se nós não passarmos os valores das duas variáveis, a função retornaria um erro, pois ela não iria conseguir realizar a soma. Para estes caso, podemos atribuir valores padrões dos parâmetros, para que ao ser inserido apenas um valor, a função consiga realizar o calculo (caso seja um dado do tipo String, ele realizara a concatenação)
* Podemos declarar uma função dentro de uma variável, criando assim uma função anônima. Exemplo:

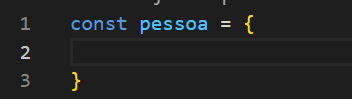


* Conseguimos declarar uma variável utilizando arrow function. Para declararmos desta forma ao invés de utilizamos a palavra function, utilizamos os simoblos => após o parâmetro. Exemplo

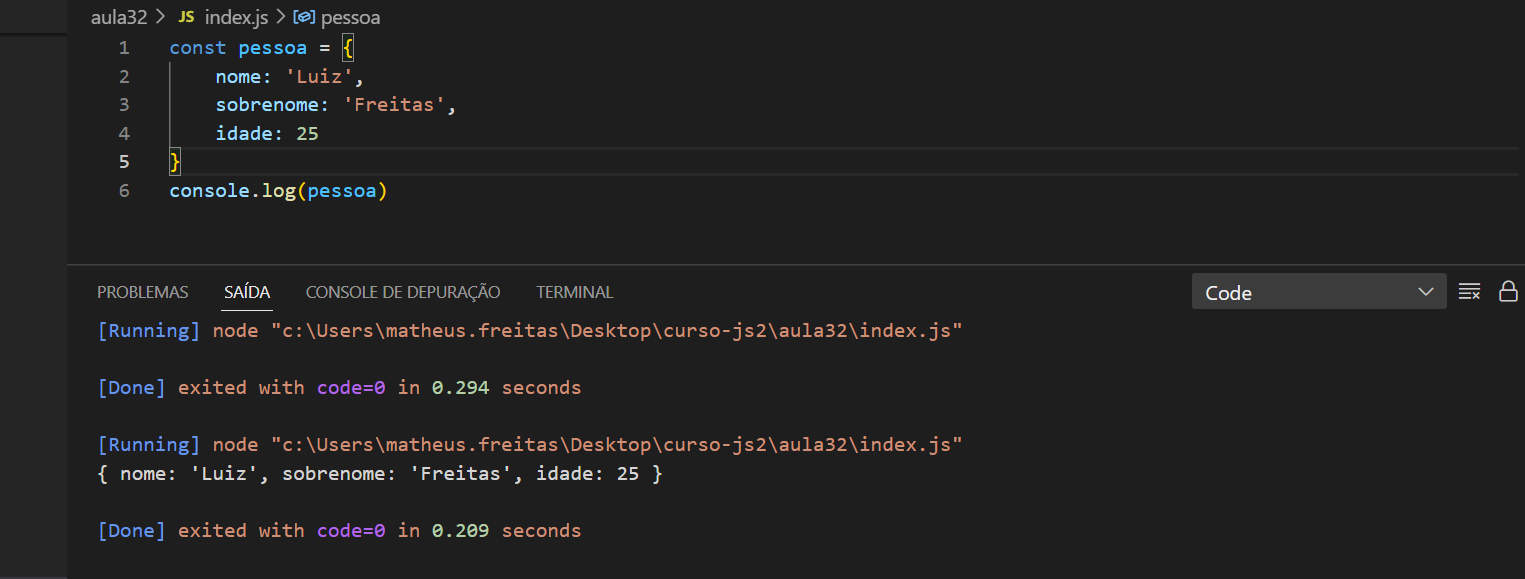


**AULA 32**

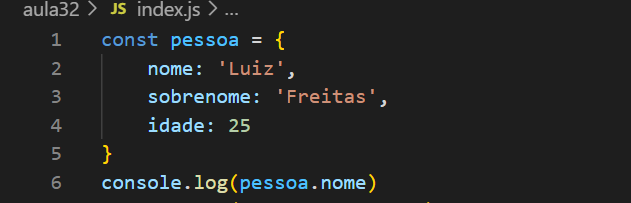
* Aula sobre objetos
* Para declarar objetos, é parecido com arrays, porém utilizamos chaves ao invés de colchetes. Exemplo:



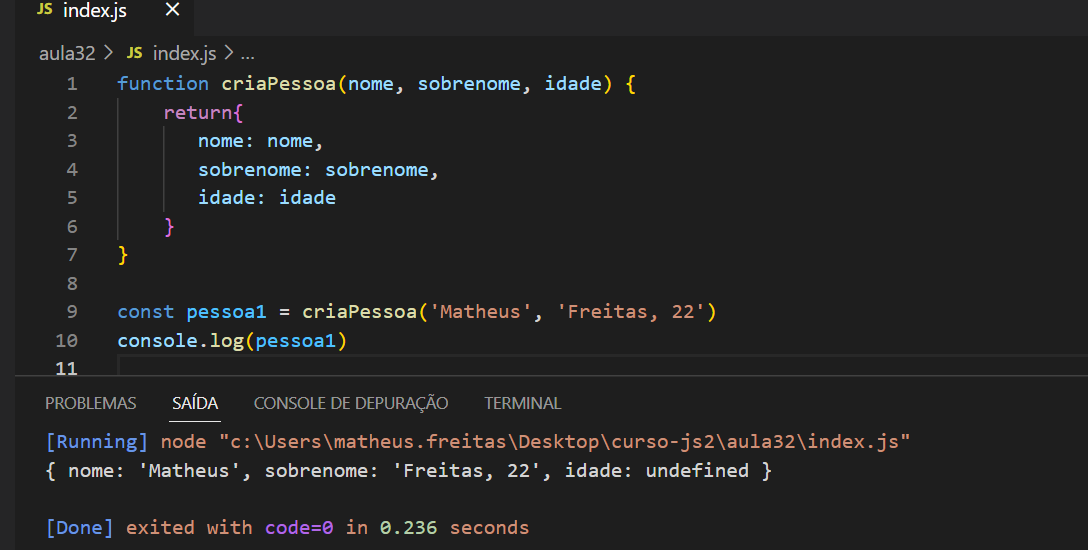
* Dentro de um objeto, podemos criar atributos, que funcionaram como variáveis dentro do objeto



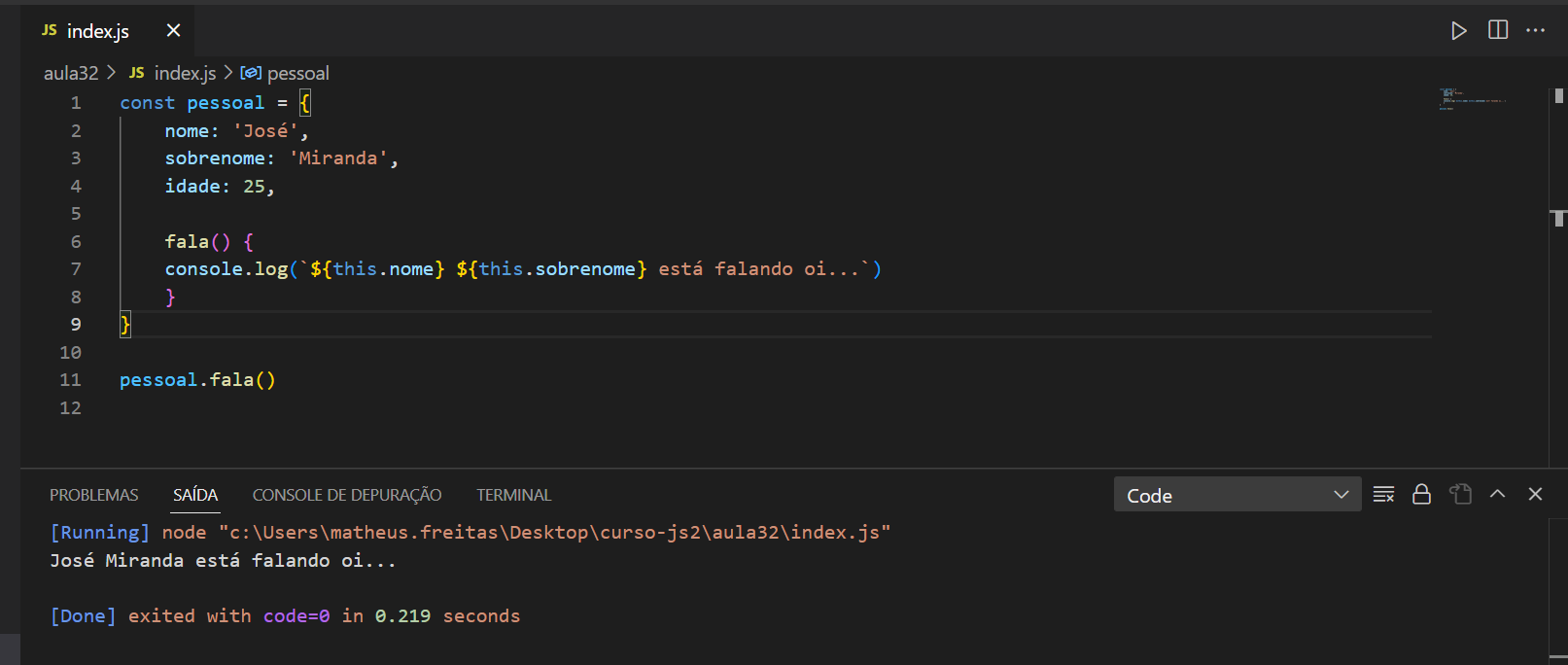
* Podemos visualizar um atributo especifico utilizando nome.atributo. Exemplo:



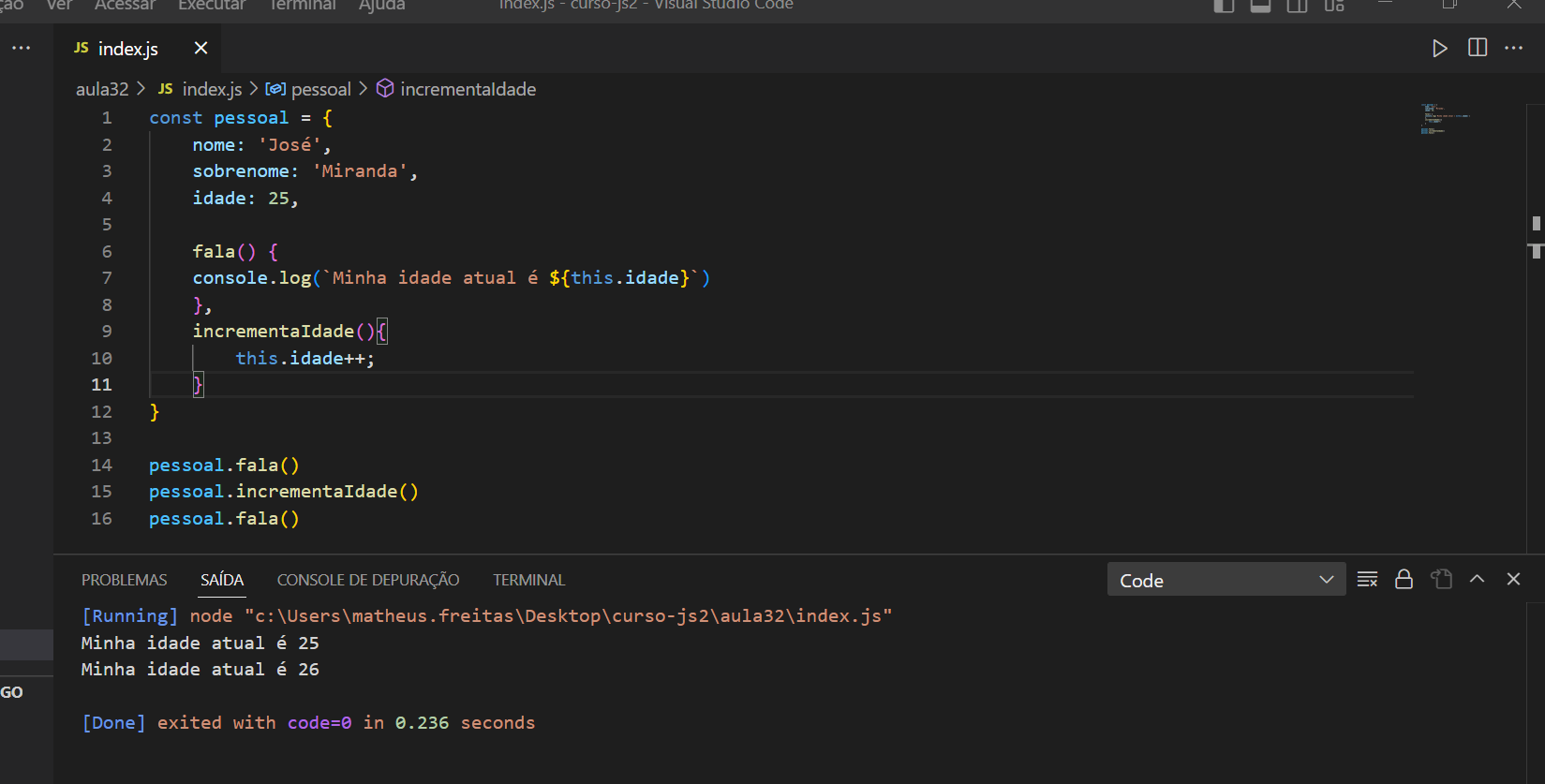
* Podemos criar uma função que crie um objeto. Exemplo:



* Podemos criar funções e métodos dentro de outras funções
* Dentro de uma função, podemos utilizar o comando this que nos permite utilizar os atributos. Exemplo:

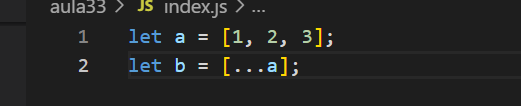


* Ele também nos permite manipular atributos da função, permitindo alterar seus valores. Neste exemplo a seguir, realizamos uma incrementação na idade do objeto toda vez que o método incrementaIdade é chamado



**AULA 33**

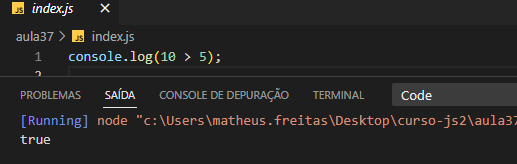
* Vimos os tipos de dados primitivos (string, number, boolean, undefined, null)
* Tipos primitivos são imutáveis
* Conseguimos alterar o valor deles, porém não conseguimos alterar os elementos de um dado, sem alterar o valor completo dele
* Vimos dados passados por referência (arrays, objetos, funções)
* Dados passados por referencia apontam para o mesmo lugar da memória, diferente dos dados primitivos
* Já este acima são sim mutáveis, pois conseguimos alterar seus elementos
* Para apenas copiarmos o valor de uma array ou objeto para uma variável, podemos fazer da seguinte forma

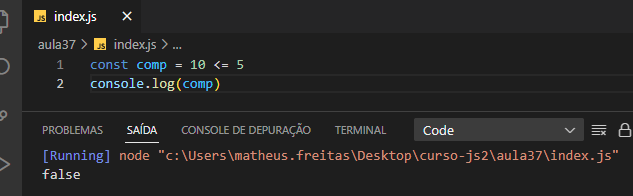


**SEÇÃO 03**

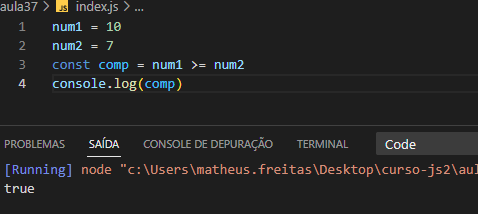
**AULA 37**

* Operadores de comparação
* Eles são representados pelos seguintes sinais:
  + > (maior que)
  + >= (maior ou igual)
  + < (menor que)
  + <= (menor ou igual)
  + == (igualdade, valor)
  + === (igualdade, valor e tipo)
  + != (diferente, valor)
  + !== (diferente, valor e tipo)
* Sempre usarmos algum desses operadores é como se estivéssemos perguntando algo a linguagem de programação. Exemplo:



Podemos criar variáveis (do tipo Boolen, que retorna verdadeiro ou falso) utilizando os operadores

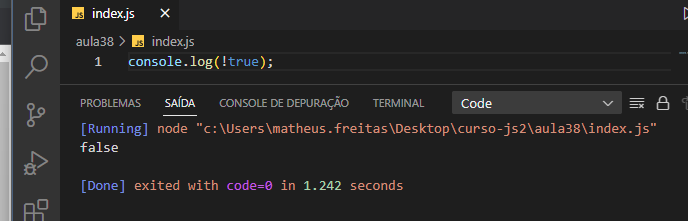
* Podemos também utilizar outras variáveis para criar uma variável boolen. Exemplo:



* O operador de igualdade (==) verifica se os dados são iguais, já o operador igualdade escrita (===) também verifica o tipo de dado (Number, String, boolean), então caso os dados sejam iguais, mas de tipos diferentes, ele retorna falso.
* Os operadores de diferença funcionam de forma semelhante ao de igualdade, sendo que:
  + != - Verifica apenas o valor
  + !== - Verifica valor e tipo

**AULA 38**

* Aula sobre operadores lógicos
* São eles:
  + && -> AND -> E – Todas as expressões precisam ser verdadeiras
  + || -> OR > OU – Apenas uma das expressões precisa ser verdadeira
  + ! -> NOT -> NÃO
* São utilizados para comparar mais de dois elementos
* O operador nega a expressão que foi retornada. Exemplo:

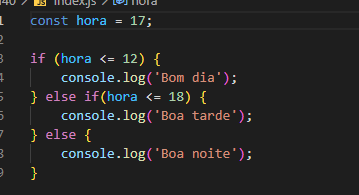


**AULA 39**

* Quando utilizamos o operador lógico && ele vai realizar a checagem, caso ele encontre um valor onde ele retornaria falso, ele nos retorna este valor, quando existem apenas valores verdadeiros, ele nos retorna o ultimo valor que ele contra
* Os valores que ele reconhece como falso são:
  + False
  + 0
  + Strings vazias (‘ ‘ ou “ “)
  + Null / undefined
  + Nan
* Qualquer outro valor que foi citado acima, é considerado verdadeiro pelo JavaScript
* O operador || vai sempre retornar o primeiro valor verdadeiro que ele encontrar

**AULA 40**

* If, else if e else
* Usados para mudar o fluxo da aplicação
* Vimos um exemplo de estrutura.

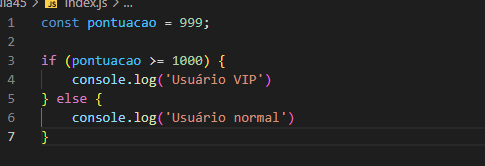


AULA 41

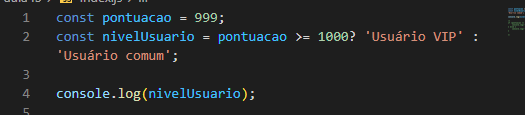
* Continuação sobre if, else if e else
* Vimos a diferença entre um código sendo executado dentro e fora do comando if

AULA 45

* Operação ternária
* Podem ser usadas para diminuir o tamanho do seu código, substituindo os comandos If e else
* Sendo assim podemos escrever o seguinte código



Desta maneira:

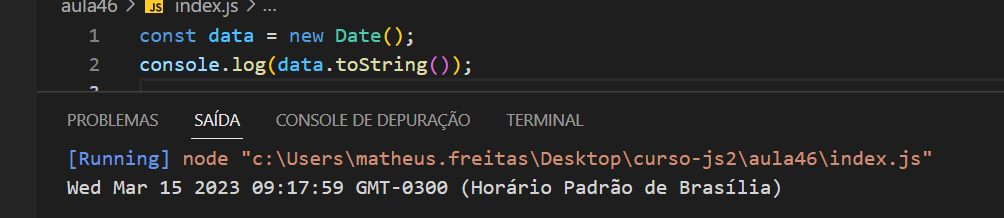


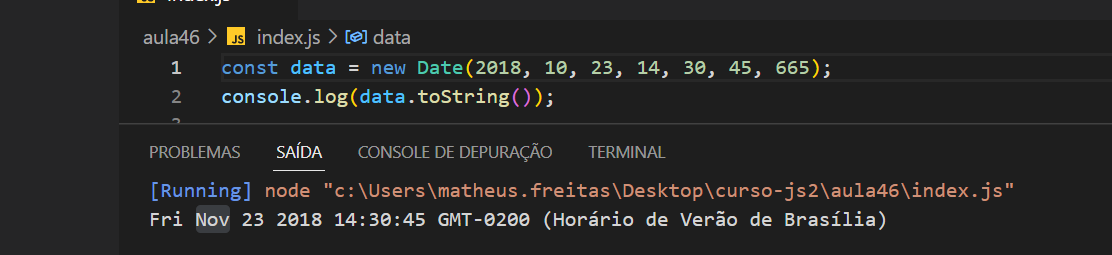
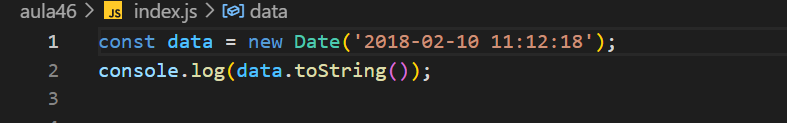
* Assim ela é composta da seguinte forma:

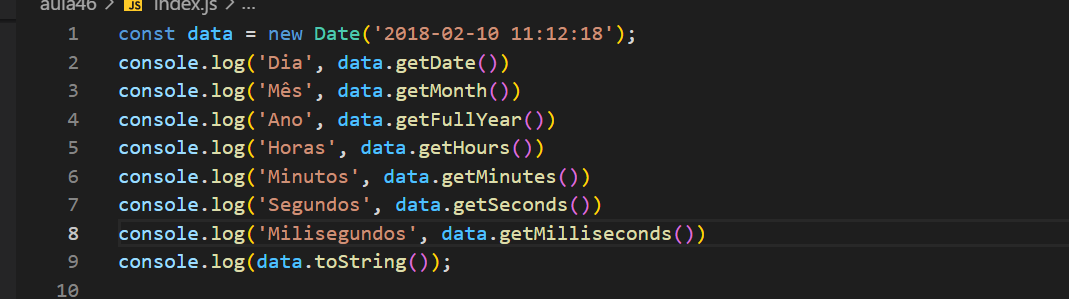
Condição ? ‘Valor para verdadeiro’ : ‘Valor para falso’;

AULA 46

* Aula sobre o objeto Date
* É um objeto construtor (um objeto construtor sempre possui a palavra new e inicia com a letra maiúscula)
* Utilizando-o, o JavaScript cria um objeto que mostra as informações da atual data, como dia, mês, hora, minuto etc.
* Podemos verificar exibir o objeto Date da seguinte forma:
* Ele utiliza o parâmetro de milissegundos, para realizar o cálculo da data
* O marco zero que utilizar para fazer o calculo é o dia 01 / 01 / 1970
* Ao todo, podemos usar quatro tipos de parâmetros ao utilizar o elemento date.
  + **Date () – Parâmetro em branco**

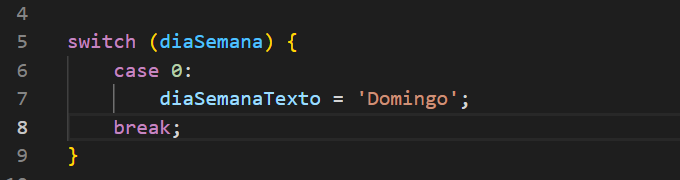


* + **Date (50) – Valor em milissegundos**
    - Podemos utilizar milissegundos para fazer cálculos com as datas, adicionando ou subtraindo tempo
  + **Date (2019, 05, 03, 14, 30, 50, 600) – Valores Ano, Mês, Dia, Hora, Minutos, Segundos e Milissegundos** 
    - Para isso, todos os parâmetros devem ser passados representados numericamente.
* 
  + **Date (‘2019-05-21 15:44:20) – Valor atribuído através de uma String**
* 
* Conseguimos extrair informações de uma dataString, como selecionar apenas a dia ou ano. Exemplo



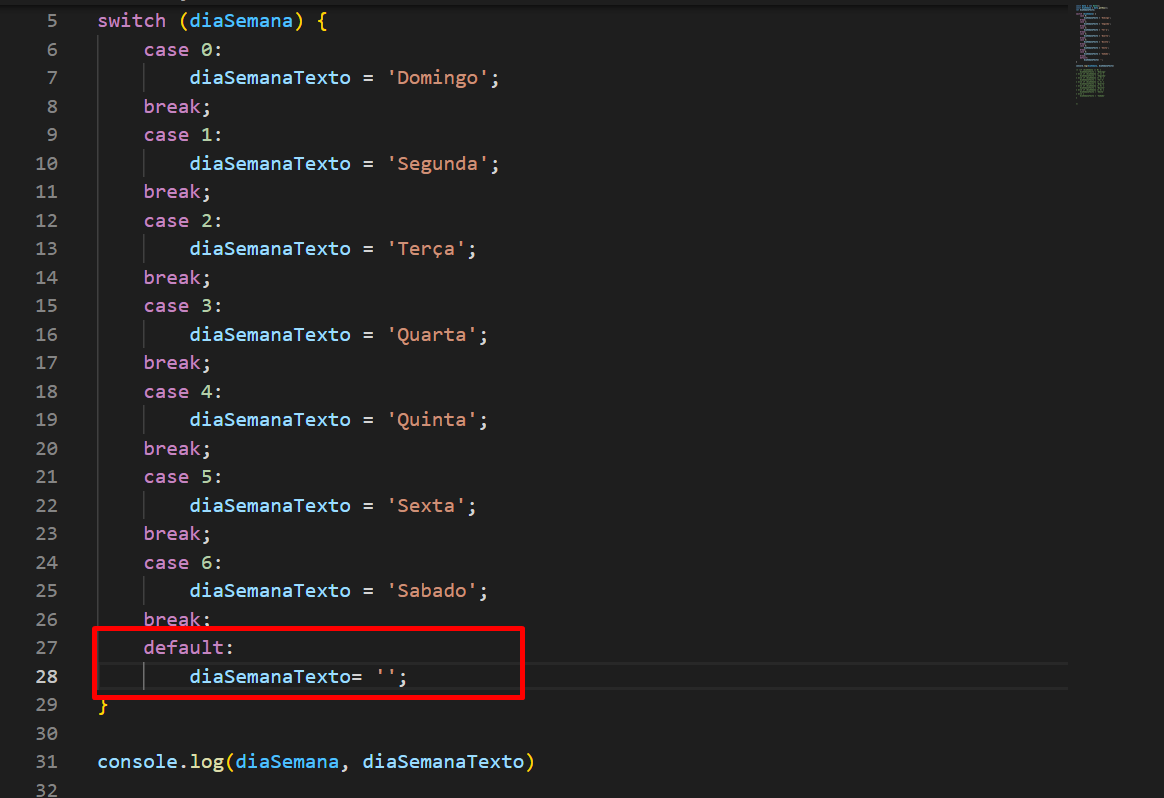
**AULA 47**

* Aula sobre switch / case
* É uma estrutura de repetição
* Pode substituir o comando if e else em algumas situações
* Podemos criar uma estrutura switch da seguinte forma:



Podemos ler como “Caso o diaSemana seja igual a zero, diaSemanaTexto recebe ‘Domingo’

* É sempre necessário finalizar um switch com o comando break sempre que se usar o comando case
* Podemos estabelecer um valor padrão para o parâmetro utilizando o comando default. Exemplo:



* Podemos utilizar essa estrutura dentro de uma variável (PS: Quando utilizada dentro de uma variável, podemos trocar break pelo comando return com gatilho de saída da repetição)

AULA 48

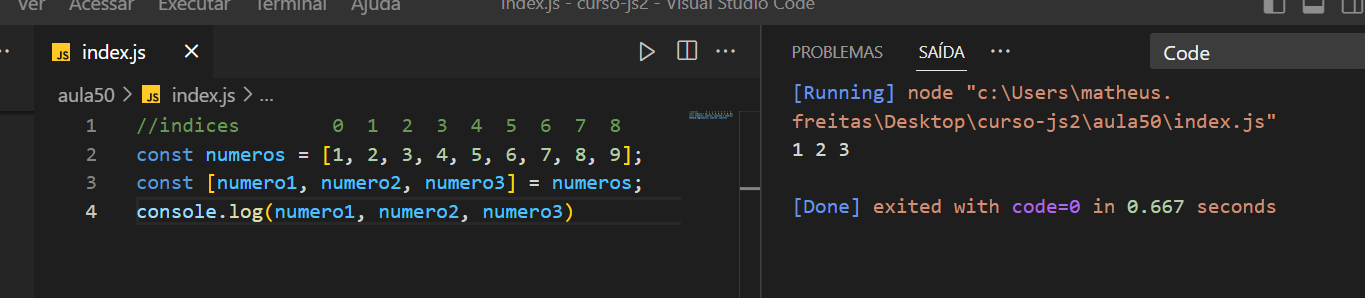
* Exercicios switch / case
* Visualizar resumo do exercício

**AULA 49**

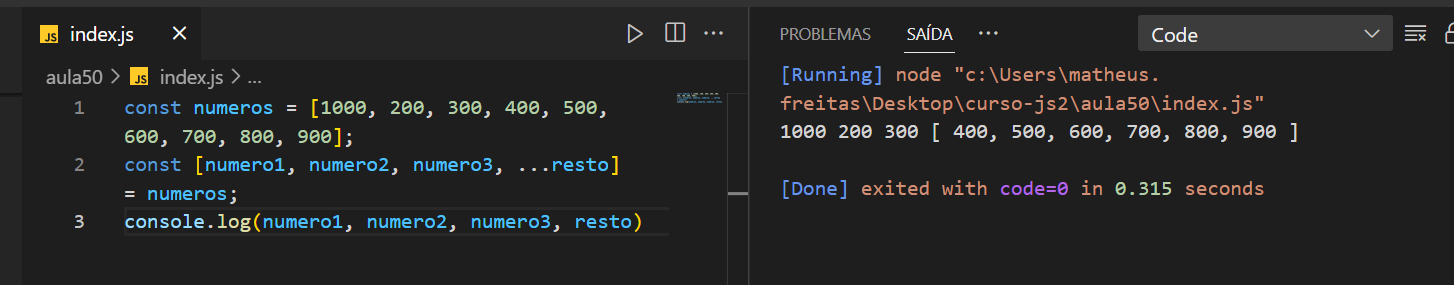
* Diferenças entre var, let e const
* Diferente de var, quando utilizamos o let para construir uma variável não podemos declarar o valor dessa variável posteriormente, exceto quando utilizado dentro de bloco (utilizado dentro de chaves, como uma função)
* Toda vez que você declarar uma variável utilizando var, como nome de uma variável já criada ele redefinira o valor da variável, independentemente de estar dentro ou não de um bloco

**AULA 50**

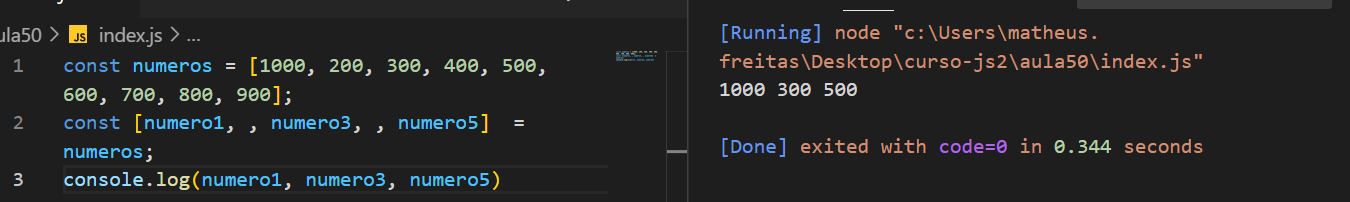
* Aula de atribuição sobre desestruturação
* Nos ajuda a economizar linhas de código, onde ao invés de criamos uma linha cada variável que queremos criar, podemos cria-la em uma única linha
* É como se criarmos uma array, onde cada índice se tornar uma variável, que quando é inicializada, recebe o valor da array indicada, onde o índice 0 recebe o índice 0 da array, seguindo dessa forma. Exemplo:



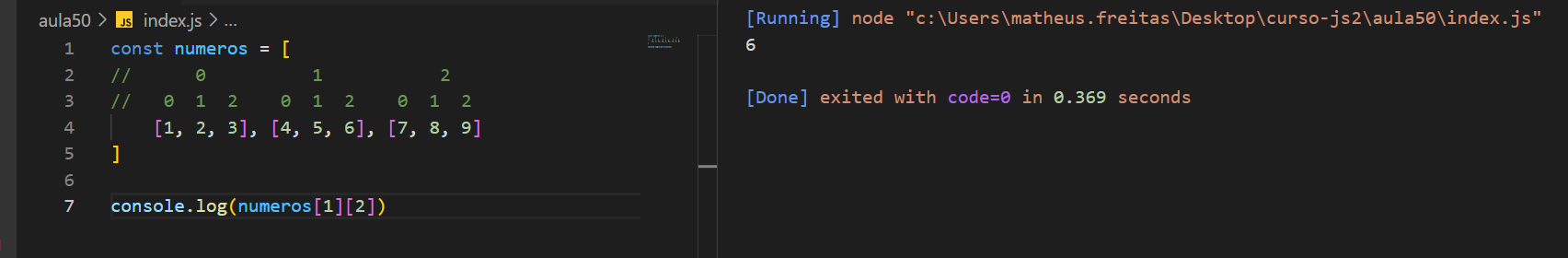
* Para os casos parecidos com a situações acima, onde ainda sobram valores do array ‘’numeros’’, podemos utilizar o comando resto para atribuir esses valores em uma outra variável que está sendo atribuída por desconstrução. Exemplo:



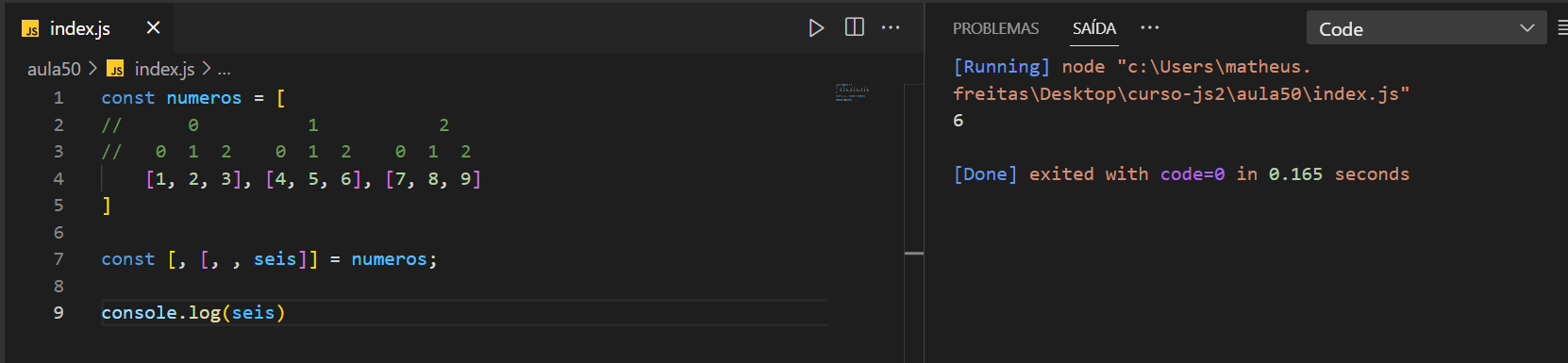
* Se por acaso quisermos pegar valores que não estão em sequencia na array, intercalando os índices, apenas precisamos deixar um campo vazio onde aquele índice seria atribuído. Exemplo:



* Podemos possuir arrays dentro de arrays, criando assim listas dentro de listas.

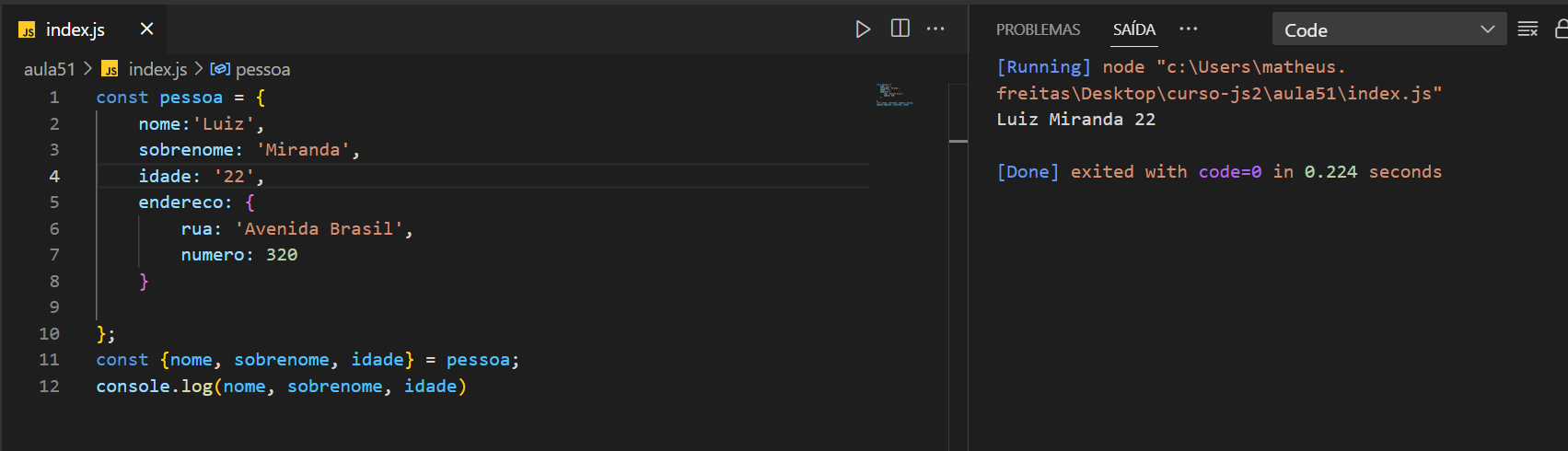


* Podemos inclusive utilizar a atribuição por desestruturação utilizando arrays dentro de arrays, usando o método dos campos vazios. Exemplo:

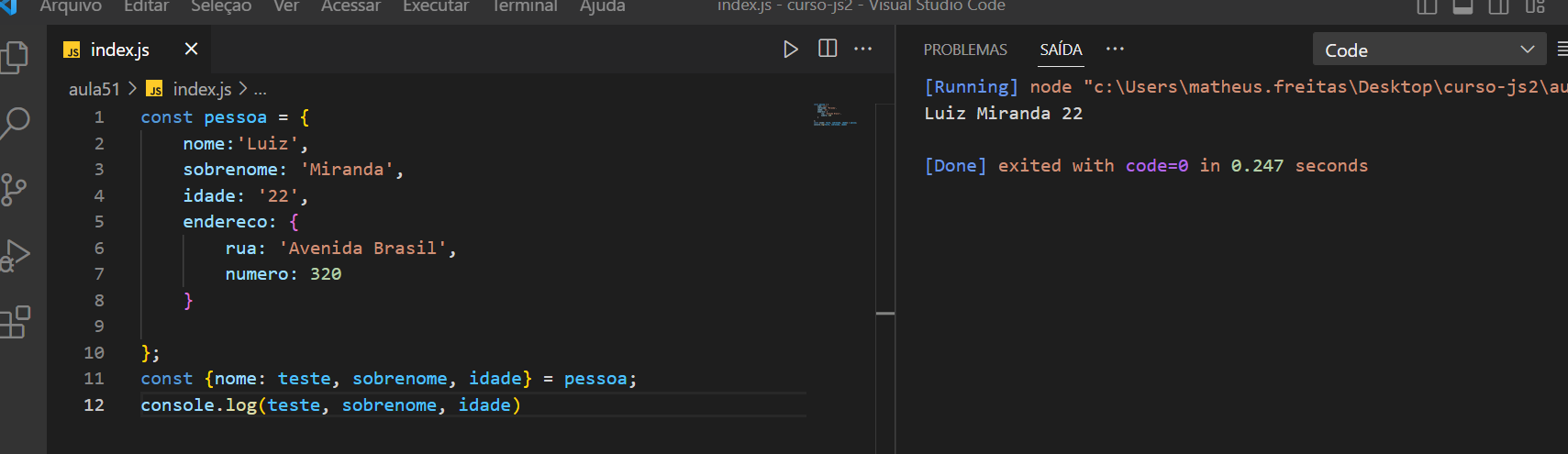


**AULA 51**

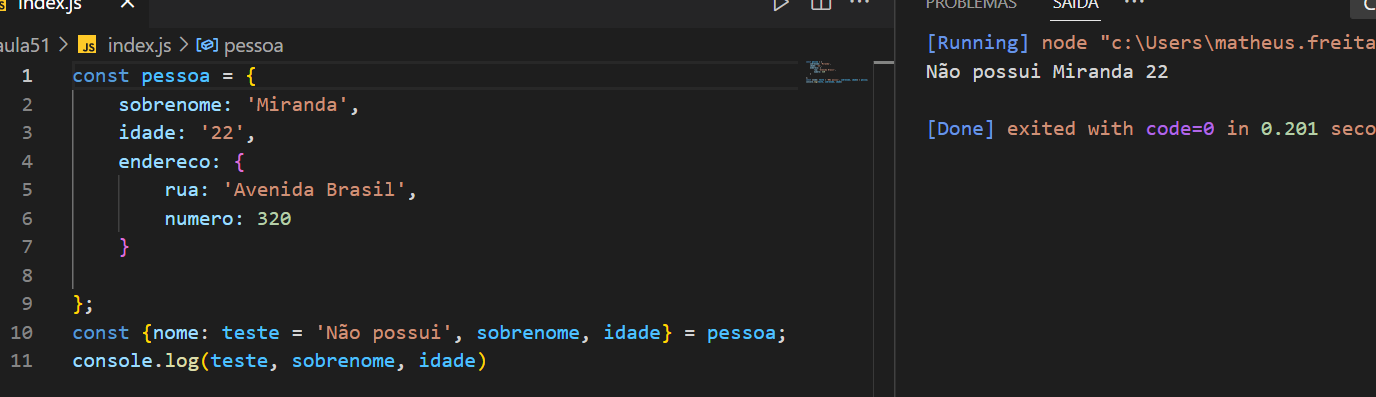
* Atribuição por desestruturação para objetos
* Para atribuir valores dessa forma com objetos, funciona de forma similar, porém ao invés de colchetes utilizamos chaves. Exemplo:



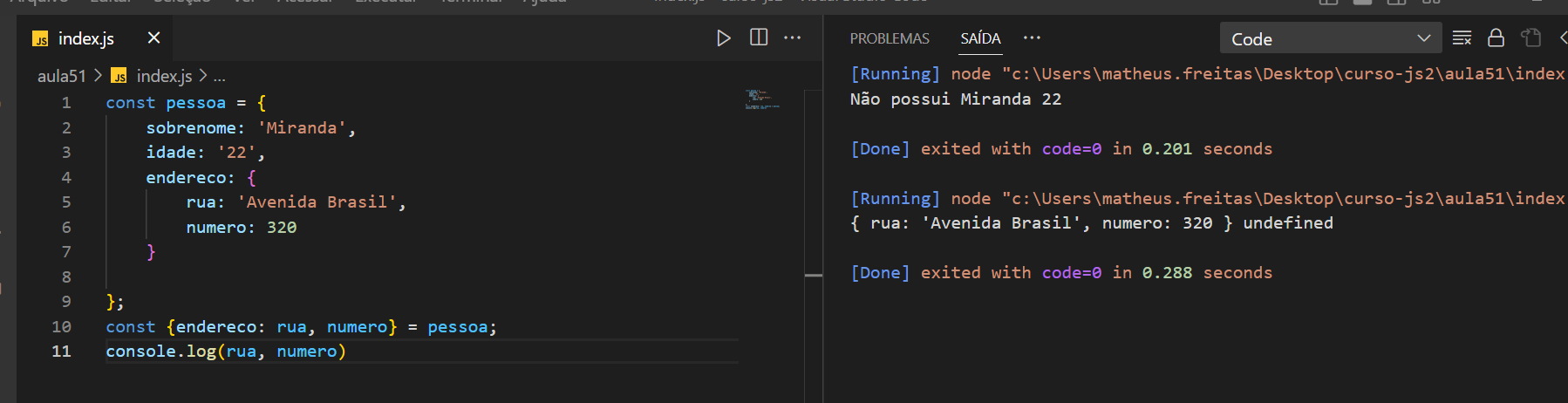
* Diferente da array, a variável precisa ter o mesmo nome do atributo dos objetos, se não o dado será apresentado como undefined
* Conseguimos manipular o nome do atributo do objeto, selecionando o nome do atributo atual e utilizando dois pontos para atribuir o novo nome, da seguinte forma:



* Podemos atribuir um valor padrão, para quando o objeto não possuir um valor no atributo selecionado. Exemplo:



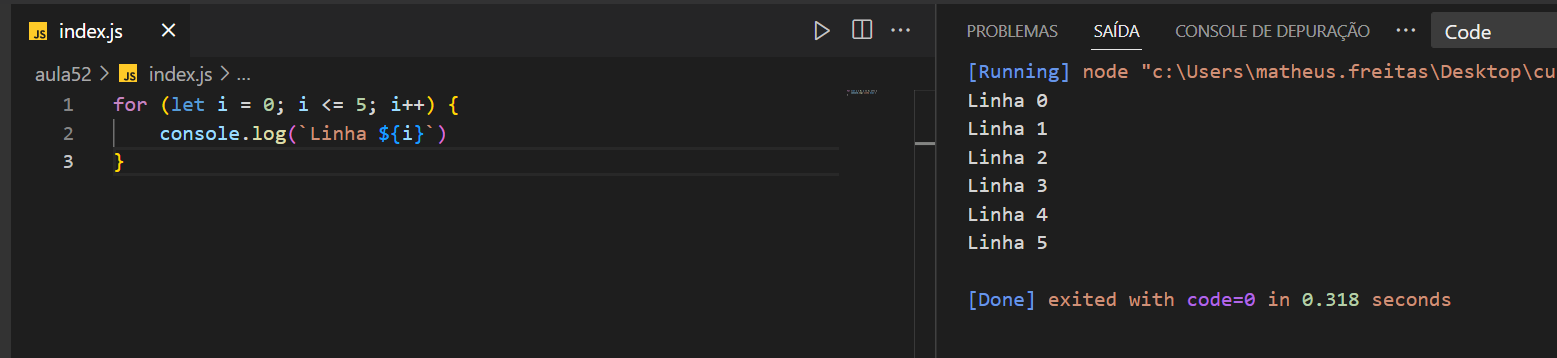
* Para selecionar um objeto que está dentro de um objeto, criamos as chaves e informamos o atributo e utilizamos os dois pontos para apresentar o atributo do objeto filho. Exemplo:



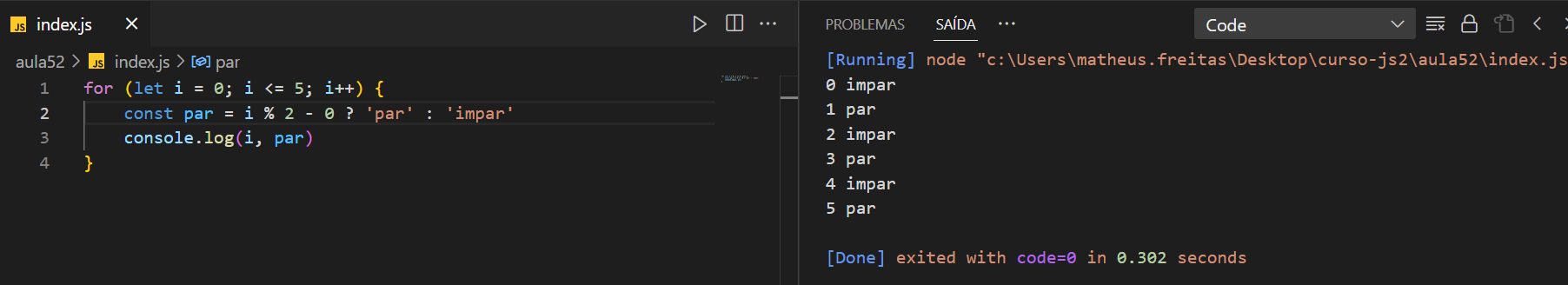
* Podemos utilizar a propriedade resto também no objeto

AULA 52

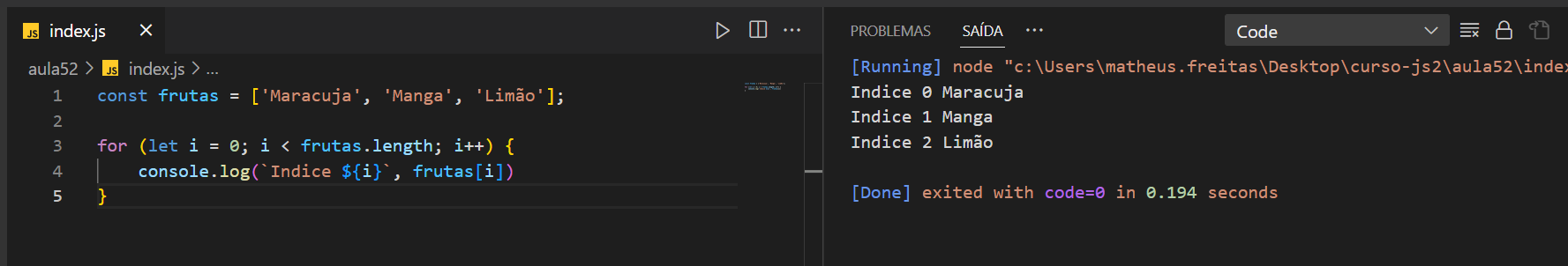
* Estrutura de repetição “For’’ (clássico)
* Para criar a estrutura, seguimos o seguinte passo a passo:
  + Declarar o comando for
  + Abrir e fechar parentes
  + Declarar variável de controle
  + Indicar condição de repetição
  + Declarar incremento ou incremento da variável de controle
  + Abrir e fechar chaves
  + Declarar conteúdo a ser repetido
* Exemplo de estrutura For:



* Exercício par ou ímpar:

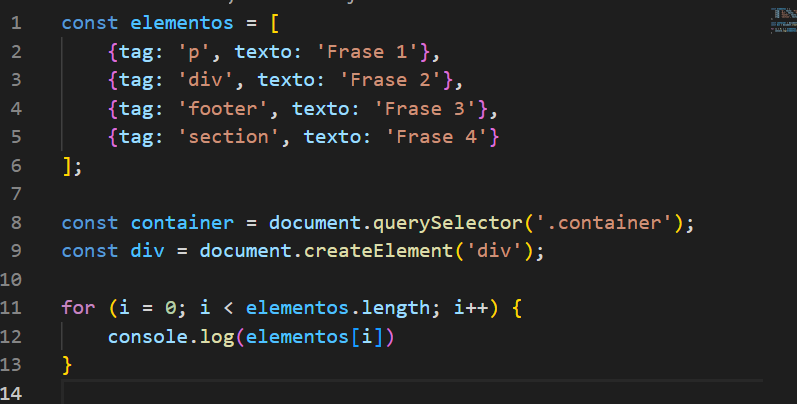


* Exercícios Frutas

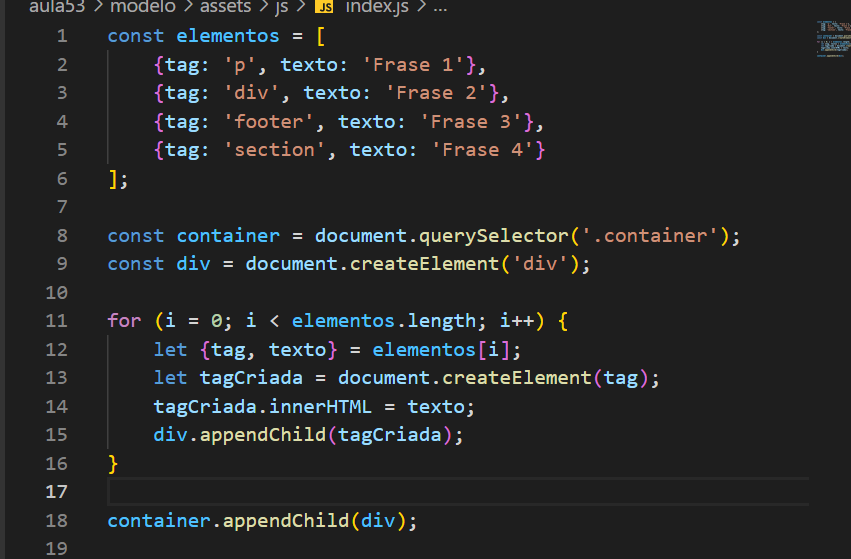


**AULA 53**

* Exercício de criação de elementos em HTML através do JavaScript
* Para criar no HTML um elemento através do JavaScript, podemos utilizar o comando createElement e para aplicar ele no HTML usamos o appendChield
* Esse comando sempre irá adicionar o elemento desejado ao final de sua caixa
* Criamos uma array que continha vários objetos dentro
* Criamos também uma const para receber o array, utilizando o campo ‘container’ criado no documento HTML
* Criamos um elemento div (mas neste momento ainda não aplicamos ele em lugar nenhum)
* Criamos um for para percorrer a array



* Dentro do for criamos uma variável e atribuímos os valores por desestruturação
* Depois fizemos com que as tags dentro da array fossem criadas, criando uma variável chamada tagCriada inferimos os valores da tags dentro dos objetos utilizando o comando createElement
* Depois fizemos a variável tagCriada receber o campo ‘texto’ do objeto utilizando o comando innerHTML
* Finalizamos o For inserindo a tagCriada dentro da Div que foi criada no JS
* Inserimos a div dentro do campo container

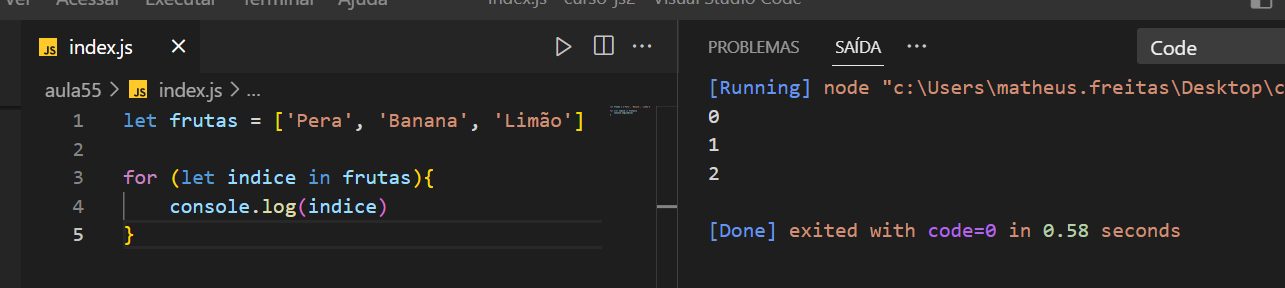


**AULA 54**

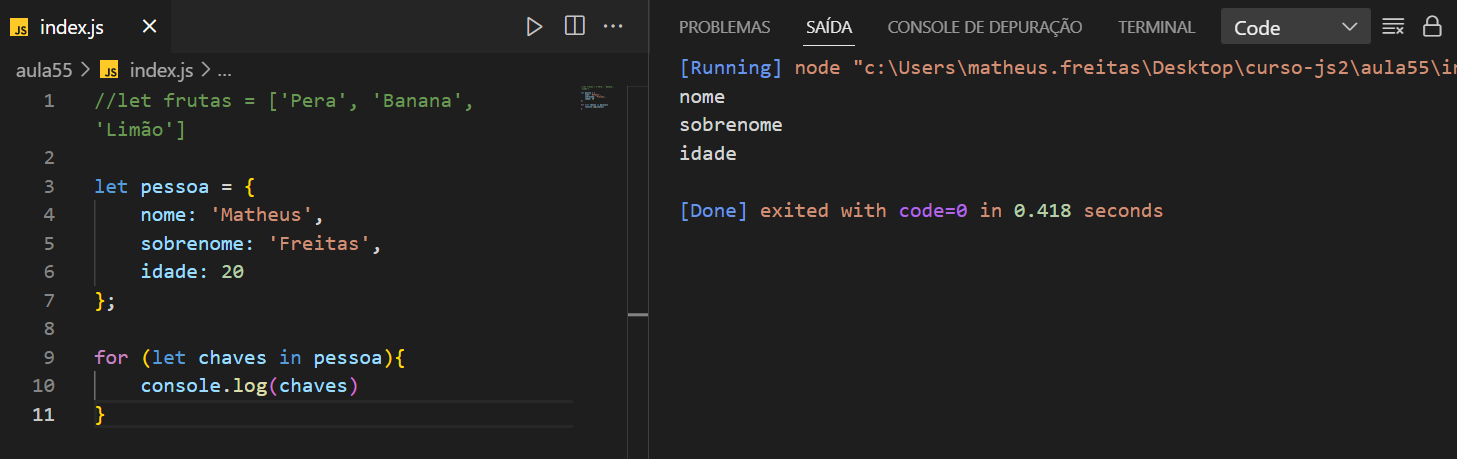
* Aula sobre a Arvore DOM
* É uma estrutura
* Possui suas hierarquias
* Window seria a hierarquia máxima, ou seja, não tem nada a se selecionar antes dele
* Dentro de Window existem muito objetos filhos, um muito utilizado é o document
* Dentro de document fica o documento HTML, então através dele conseguimos manipular o head e o body, e dentro desses dois os seus elementos e objetos filhos
* Os elementos “pais” chamados de “parent” e os elementos filhos chamados de ‘’child’’
* Quando utilizamos o comando appendChild, estamos dizendo dentro deste campo, insere um elemento filho, informando qual será o elemento filho

**AULA 55**

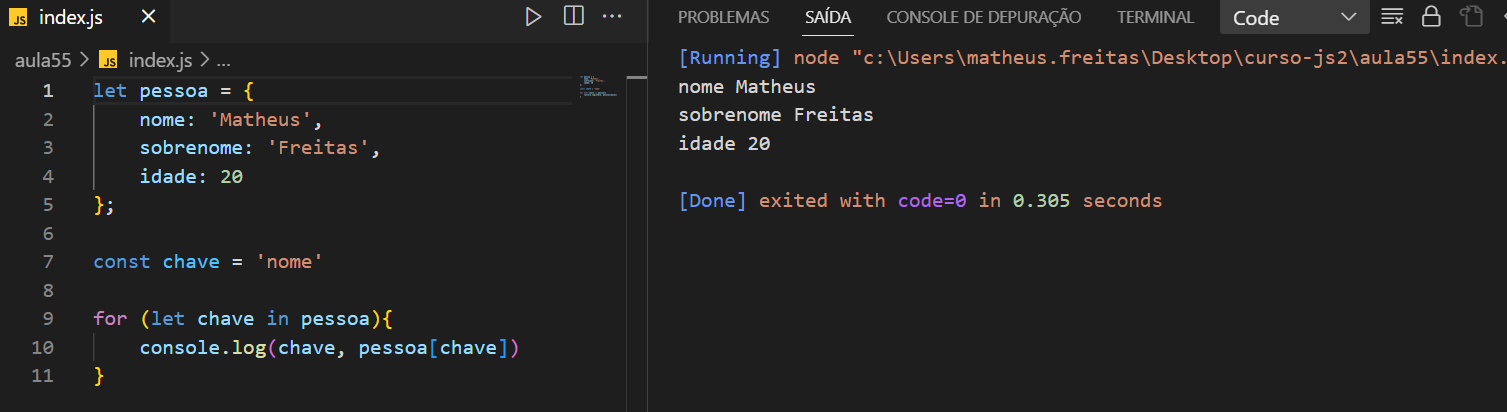
* Estrutura de repetição For in
* Realiza a leitura dos índices ou chaves de uma array ou objeto. Exemplo:



* Exemplo utilizando um objeto:

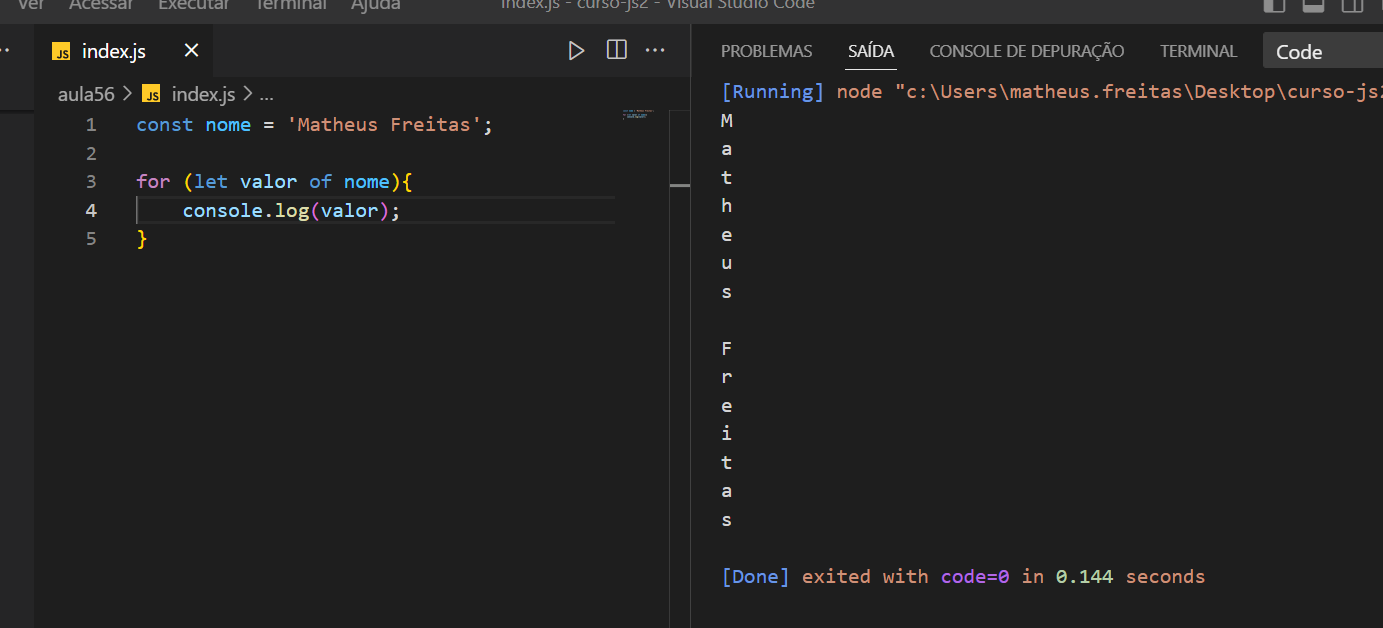


* Dentro do For In podemos utilizar variáveis como índice ou chave de uma array ou objeto. Exemplo:

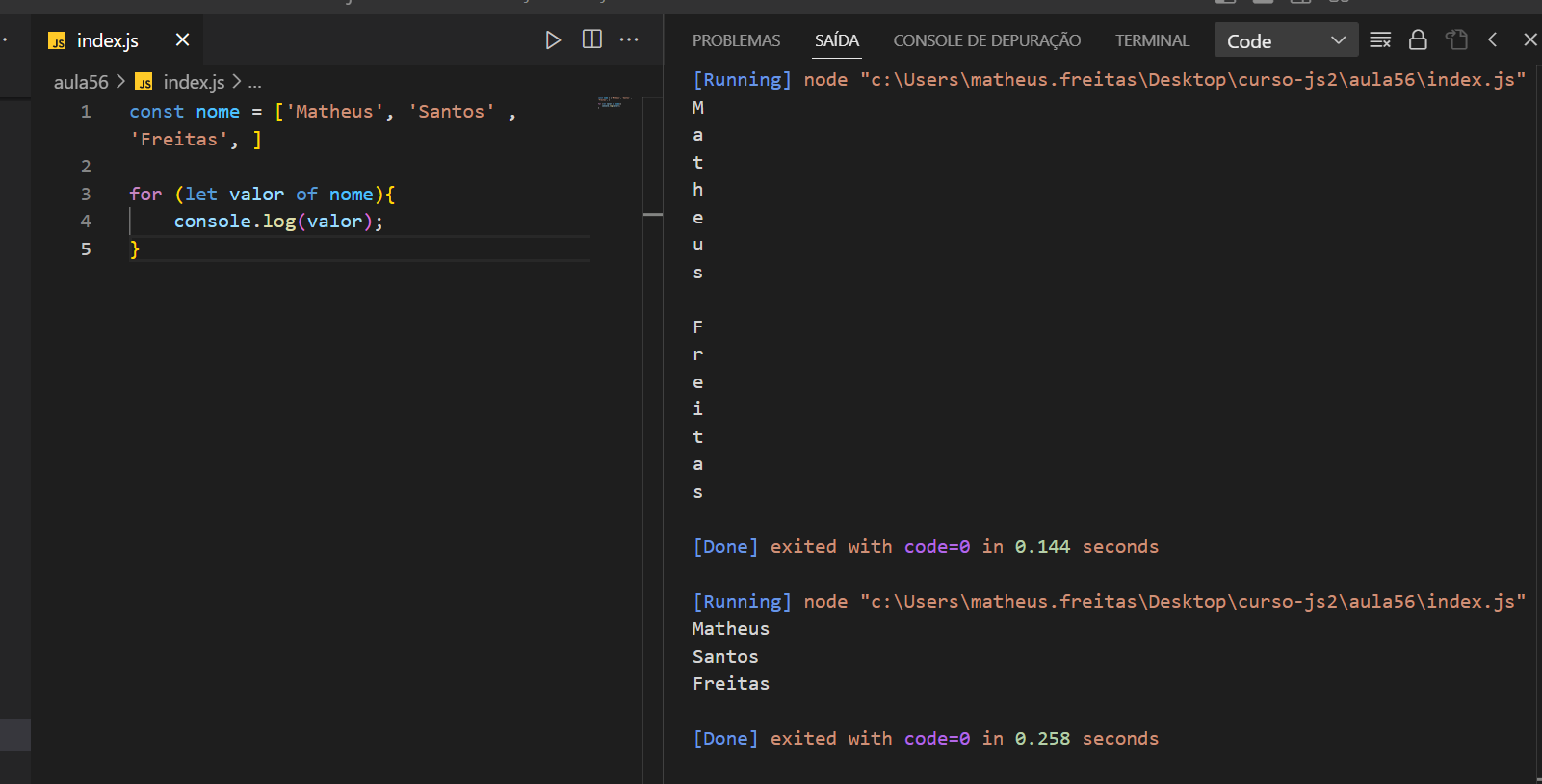


**AULA 56**

* For of
* Estrutura para elementos com inteiráveis (elemento que possuem índice)
* Funciona de forma semelhante ao for in, porém ao invés de ler o índice ele realizar a leitura do valor



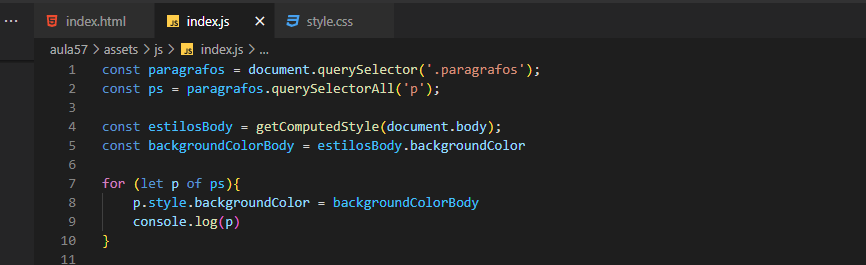
* Funciona também em Array



* Não funciona com objetos

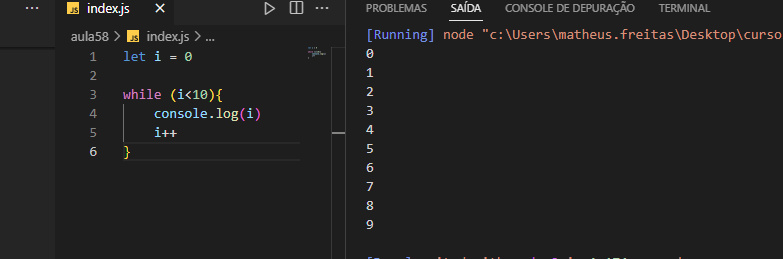
**AULA 57**

* Exercícios
* Criamos uma Nodelist (que funciona de forma silimiar a uma array)
* O comando querySelector seleciona o primeiro elemento com o id ou classe selecionado
* Para selecionar um conjunto, utilizamos o comando querySelectorAll
* Criamos uma variável utilizando o DOM para capturar o estilo css
* Aplicamos o estilo css capturado utilizando o DOM

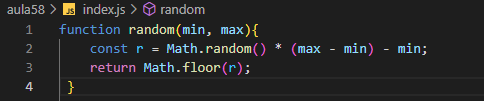


**AULA 58**

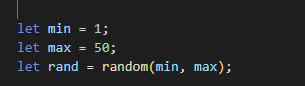
* Estrutura While e Do While (enquanto)
* Utiliza uma variável criada fora da estrutura
* Possui também uma condição para a repetição do laço



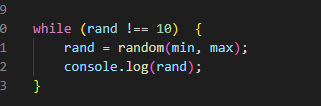
* Criamos um algoritmo que gera números aleatórios que se encerra quando encontra o valor 10
* Primeiro criamos uma função chamada Random, onde ela retorna a variável r que possui o valor de uma equação:



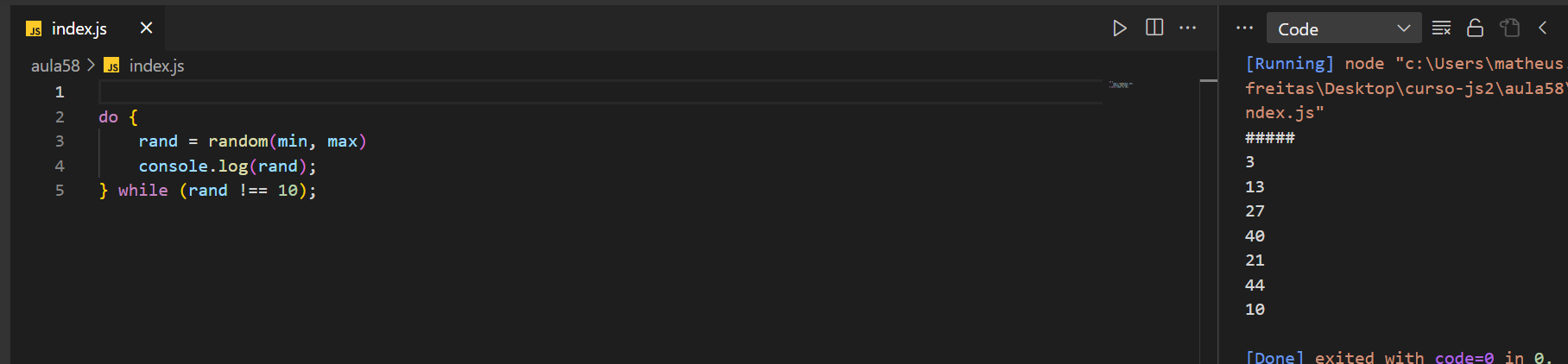
* Depois estabulemos os valores de min e max em uma variável e criamos uma variável que atribui os valores a função



* Depois criamos a estrutura White, que cria o laço de repetição

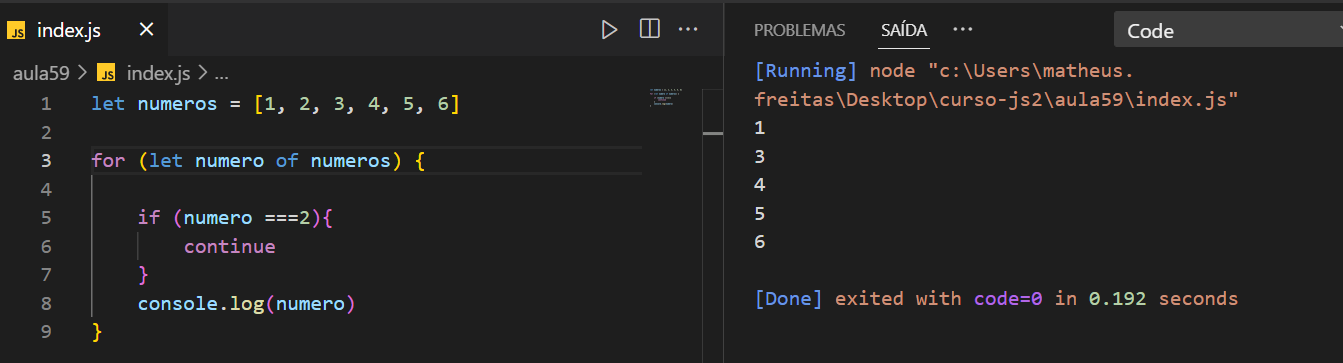


* O comando While executa o código e depois checa a condição, dirente da estrutura Do While que se checa primeiro a condição e se a condição for falsa, ela não executa o código
* Estrutura Do While

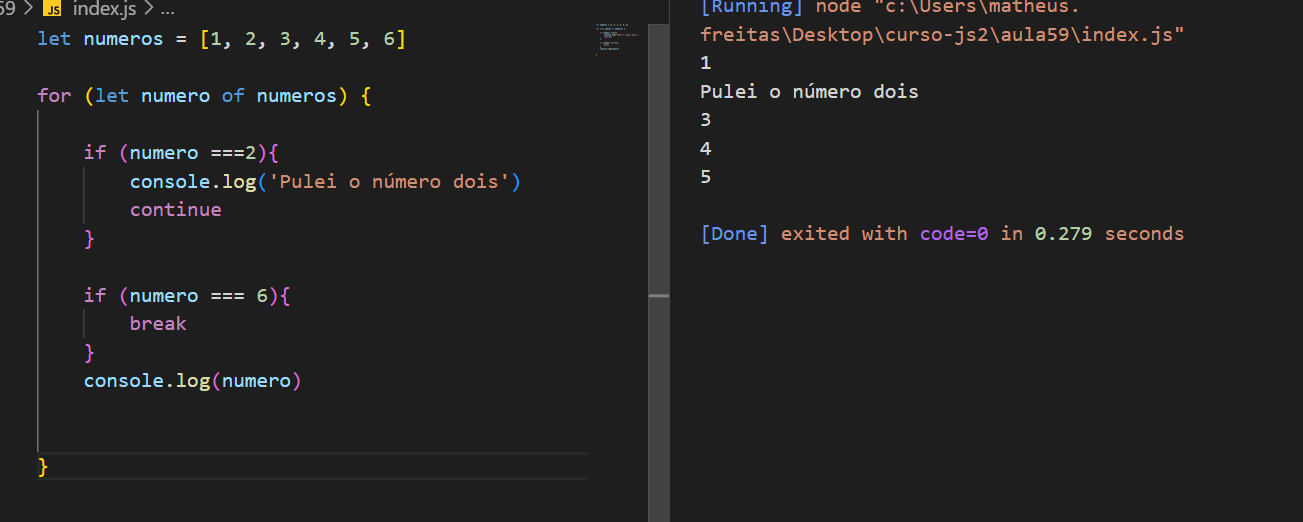


AULA 59

* Comando continue
* O comando continue é um comando que encerra o laço de repetição, fazendo com que ele pule a ação e passa para o próximo laço, fazendo com que não tenha ação durante aquele ciclo seguindo a condição escolhida. Exemplo:

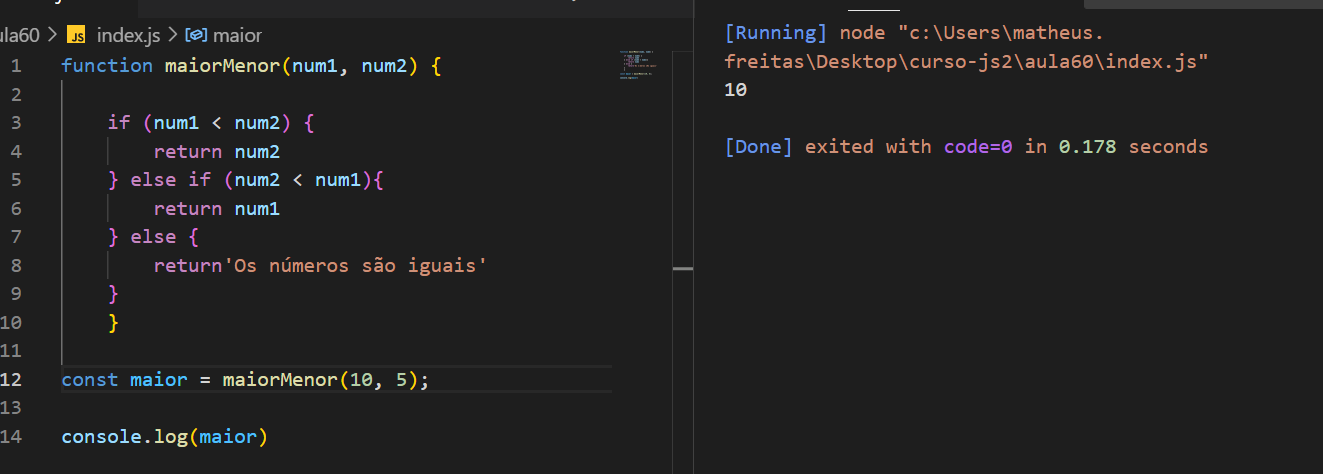


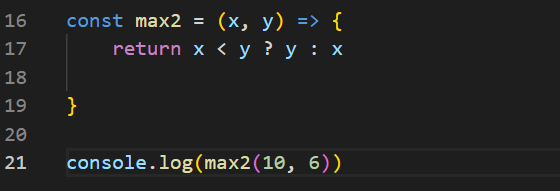
* Tudo que ficar abaixo do continue será executado apenas no próximo laço
* Comando Break
* O comando break, também encerra o laço de repetição, como o comando continue, porém ele encerra todos os laços de repetição, fazendo com que a estrutura pare de ser executada



* Os dois comandos funcionam com For, For In, For Of, While e Do While

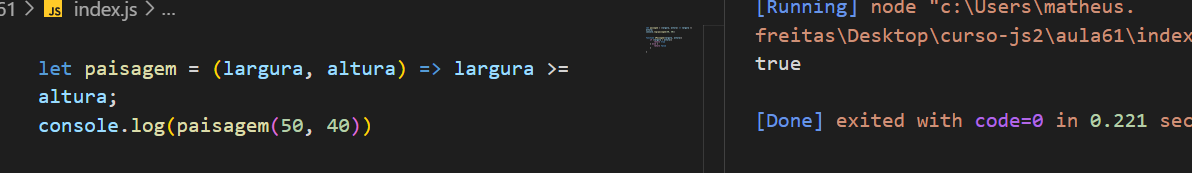
**AULA 60**

* Exercício – Criar uma função que recebe 2 números e retorne o maior deles
* A forma mais fácil de se resolver o exercício é assim 
* Também podemos representar ela dá forma ‘’Arrow’’



**AULA 61**

* Exercício – Criar um algoritmo que receba altura e largura e retorne se é verdadeiro ou falso se a orientação de uma tela estiver no modo paisagem
* Resposta utilizando arrow



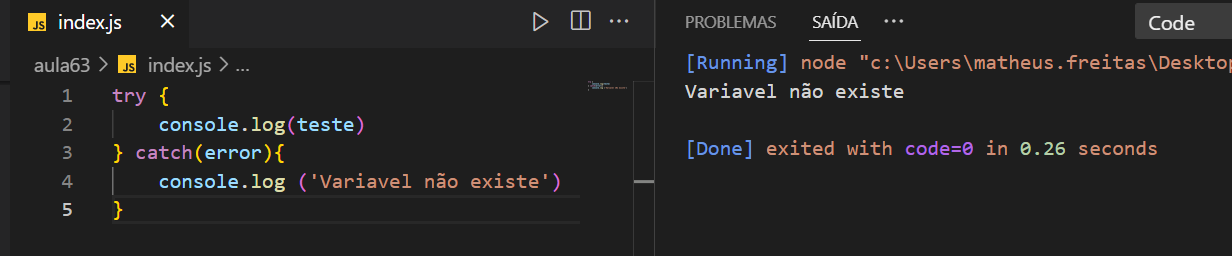
**AULA 62**

* Exercício FizzBuz – Criar um exercício que apresente os números de 0 a 100 e aqueles que são divisíveis por 3 e 5, apresentarem a menagem Fizz Buzz, apenas 3 Fizz e apenas 5 Buzz

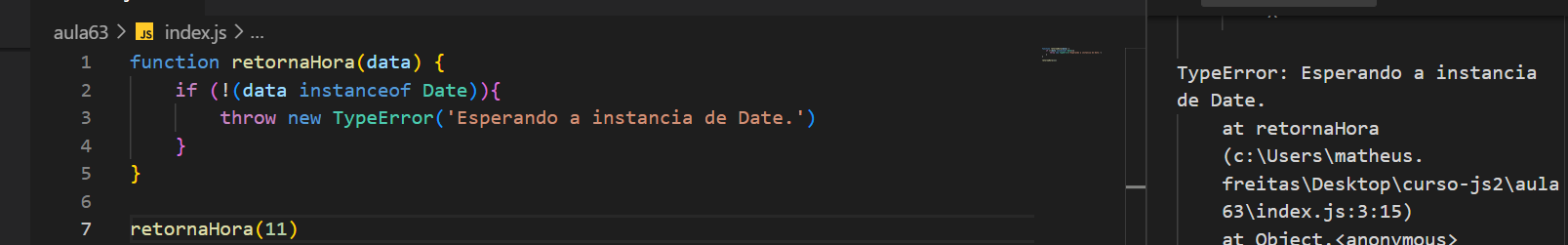


**AULA 63**

* Existem comandos que podemos utilizar para fazer teste com novos códigos, entre eles temos os comandos try e catch
* No comando try {} inserimos o nosso código que desejamos testar
* No comando catch inserimos uma variável que guarda o erro e uma ação que deverá ser feita caso o erro se concretize
* Exemplo:



* Também temos o comando throw, utilizado para “cadastrar” um erro e criar uma nova categoria

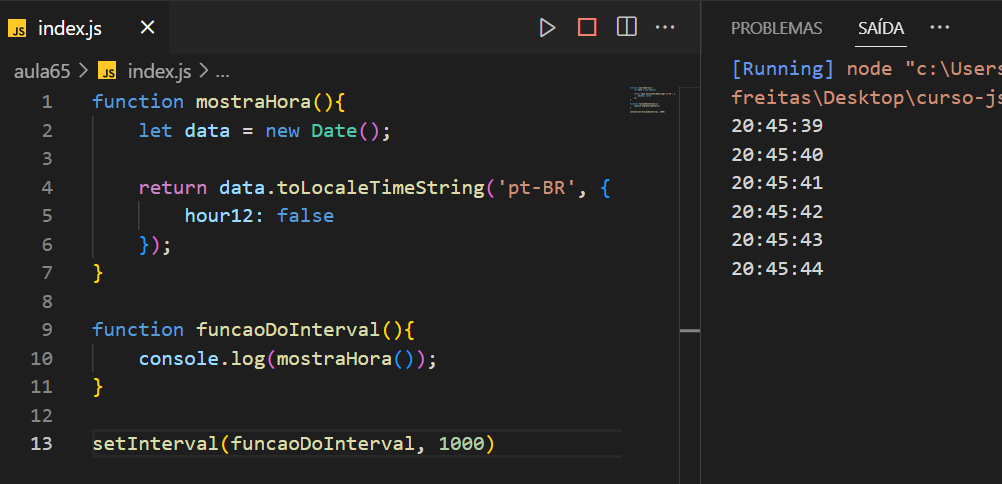


**AULA 64**

* Comando finally
* Vai após o comando catch
* Sempre é executado, dando erro ou não, sempre ao final do código de teste

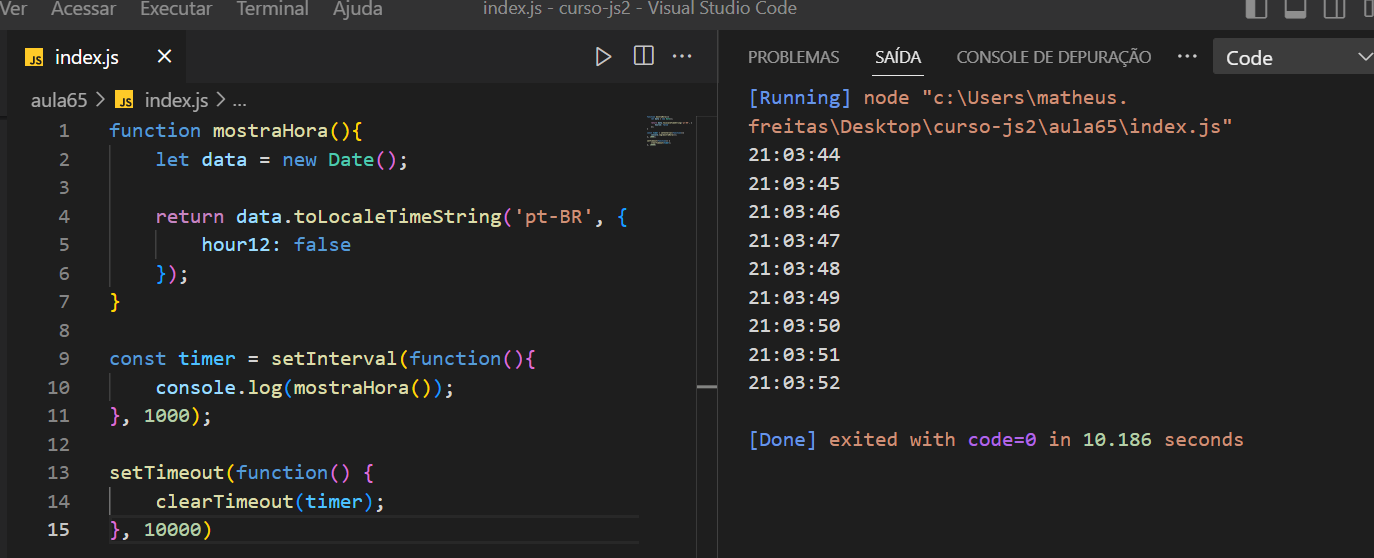
**AULA 65**

* Duas funções usadas para controlar times
* O comando setInterval executa um outro determinado comando em intervalo de tempo que estabelecemos. Exemplo:



Neste exemplo, configuramos o comando para executar a cada 1000 milissegundos a função funcaoDoInterval

* Para encerrar a execução do código, pressionamos as teclas “Ctrl” + “shift” + “p” e digitamos “stop runner code”
* Também podemos utilizar o comando clearInterval para parar a ação de de repetição. Exemplo:



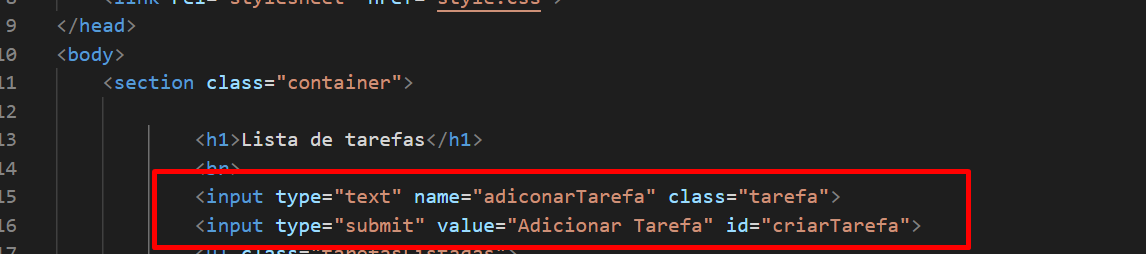
Neste exemplo, criamos uma função anônima dentro da variável timer, que realiza a repetição a cada 1 segundo e criamos o comando setTimeout, também utilizando uma função anônima com o comando clearInterval para parar a contagem da variável timer

**AULA 66**

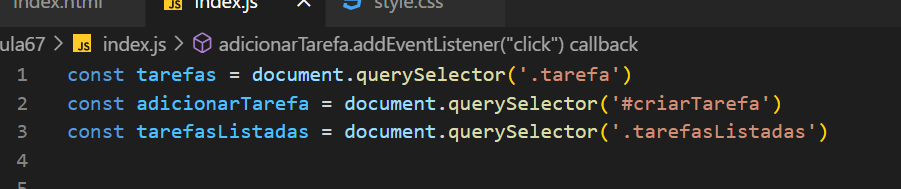
* Criamos um timer

**AULA 67**

* Exercício – Criar uma lista de tarefas onde é possível adicionar ou remover itens da lista de tarefas
* Criamos um campo para receber as tarefas e um botão para inserir elas na lista



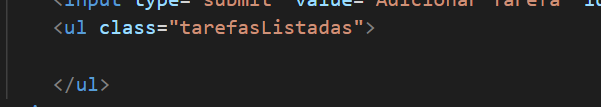
* Adicionamos a eles o ID e criamos variáveis no JS para conseguir manipular esses campos



* Capturamos o evento de click ao clicar no botão para enviar as tarefas e pausamos o envio



* No HTML criamos uma lista não ordenada e aplicamos uma classe para ela



* No Java Script, criamos uma função que cria itens de lista e acionamos ela em uma segunda função, que aciona a primeira e insere o texto adicionado no campo input na lista não ordenada

