**PROVA AVALIATIVA JAVA SCRIPT**

Raíssa Antonia Bernardo de Oliveira

2TDS

a) A DOM é como uma ponte entre o JavaScript e o HTML. Ela transforma o site em uma estrutura, fazendo com que o JS consiga acessar, mexer e controlar tudo que tá na página web. Como em textos, estilos, botões e etc.

Um exemplo é o document.getElementById(“”). O DOM é “ativado” pelo document e acessa o que estará entre aspas, podendo fazer alterações.

b) O Java Script é uma estrutura que transforma o HTML “seco” em uma página interativa. Um bom exemplo é o corpo humano.

O HTML é o esqueleto, o css é o “jeitinho” do corpo (cor de pele, estatura, etc) e o JS sãos os músculos, que permitem os movimentos e interações com outro corpo.

c) Eu gosto de pensar que o ‘==’ é mais “relaxadão” e o ‘===’ é mais “rígido”. Isso porque, o == faz a conversão de valores, se for necessário. Exemplos:

‘1’ == 1 🡺 aqui, a resposta vai ser true, porque o JS vai entender que um é uma String e vai converte-la para int, correspondendo que 1 é igual a 1.

Aqui, já enfrentamos uma problemática. Por que há uma inconsistência de dados.

‘1’ === 1 🡺 aqui, a resposta vai ser false, porque não haverá conversão de tipos. Logo, uma String não pode ser igual a um inteiro.

d) Duck Typing é quando apontamos uma variável pelo que ela pode fazer e não pelo tipo dela. Um exemplo muito usado é o do pato: “Se anda como um pato e grasna como um pato, logo, é um pato.”.

2-

a) O Arrow Function é uma forma de codar de maneira mais curta, mais direta. Ela pode ser útil para reduzir tempo de trabalho.

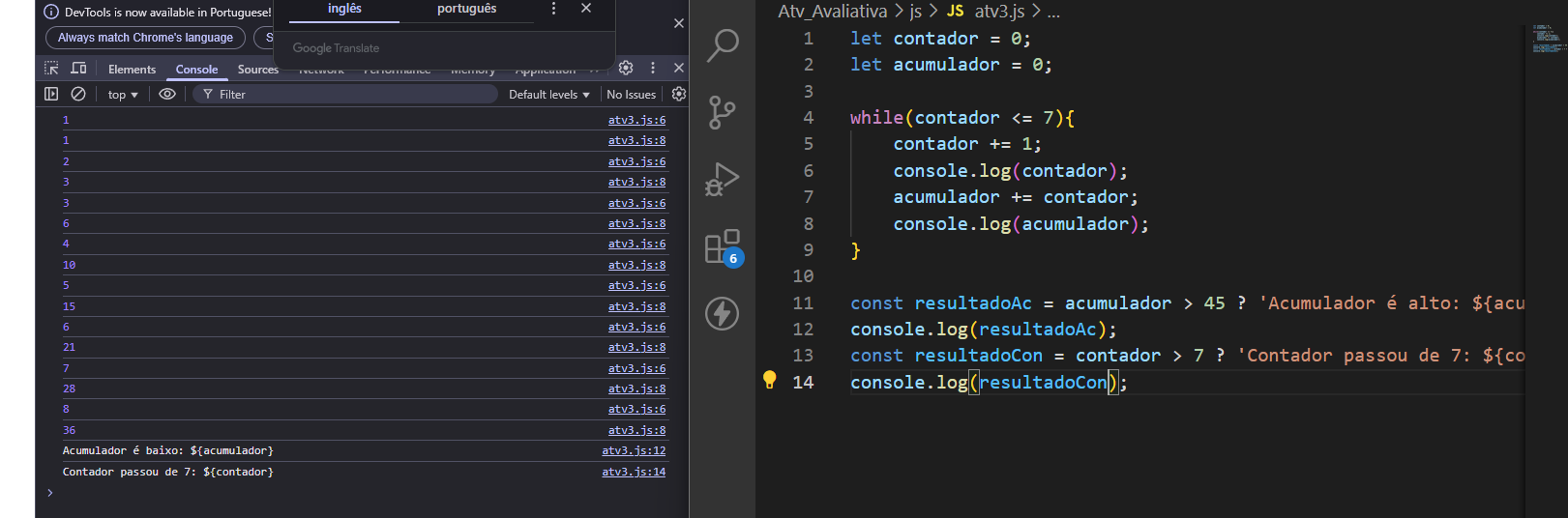
b) Uma função CallBack tem a finalidade de encurtar e facilitar o código. Nela, nós usamos uma função como parâmetro de outra função.

Como exemplo, podemos considerar um código onde em todos os botões, haverá uma mensagem escrito “Você clicou no botão”. Desse modo, podemos criar a call-back com essa mensagem e chama-la em todos as funções de botão.

c) As propriedades públicas, podem ser vistas e acessadas por qualquer pessoa. Já quando elas estão privadas, podem ser acessadas somente dentro do próprio código. Em Java Script, o encapsulamento pode ser usado para esconder partes do objeto que não deveriam ser acessadas por qualquer pessoa, prendendo-as no código.

d) TESTE DEPOIS

3-



Temos essa saída, pensando que o acumulador é baixo e o contador é alto. Porque , apesar do console não ter impresso os resultados, quando damos um console.log na estrutura de repetição (for), vemos que o contador sobe até 8, mas o acumulador não passa de 36.