Lista de Exercício - JS

Professor: Leandro Melo

Matéra: Tecnologia para Front-End

Aluna: Helena Trombetta

Matrícula: 2025211880176

Questão 1 - Um programador está desenvolvendo um pequeno script em JavaScript para exibir uma mensagem personalizada ao usuário. Ele escreveu o seguinte código:

```
<script>
let nome = "Leandro";
let idade = "45";
let anoAtual = 2025;
// Convertendo a idade para número
let idadeNumero = idade - 0;
let anoNascimento = anoAtual - idadeNumero;
if (idadeNumero > 30) {
document.write("Olá, " + nome + "! Você nasceu em " +
anoNascimento + ".");
} else {
document.write("Olá! Idade não informada corretamente.");
}
</script>
```

---

a. Qual será a saída exibida na tela? Explique o motivo.

- Olá, Leandro! Você nasceu em 1980.

- Pois, o document.write puxa o texto que precisa printar, a partir das variáveis declaradas + as expressões estabelecidas.

b. No código, acontece alguma conversão automática de tipos? Se sim, identifique onde.

- Sim, há uma conversão de tipos de variáveis. Quando ele declara: let idadeNumero = idade - 0

c. Caso a variável idade fosse declarada assim: let idade = 45; O resultado mudaria?
Justifique.

- Não, o que mudaria seria o trabalho que o software não teria mais, pois não existiria a necessidade da transformação da string em número, explicada acima.

d. O que aconteceria se o programador trocasse let por var nas variáveis? Isso poderia gerar
algum problema no escopo? Explique.

- O var tem escopo global, dito isso fora da possível função. Nada aconteceria de imediato, só traria problemas caso ocorra uma redeclaração, let e const são melhores práticas.

Questão 2 - Um programador escreveu o seguinte script com várias operações encadeadas. Analise detalhadamente o código e responda às perguntas.

```
<script>
var x = "10";
let y = 5;
function calcular(a) {
var x = a + y;
if (x == "15") {
x = x + 5;
}
return x;
}
let resultado = calcular(10);
switch (typeof resultado) {
case "string":
resultado = resultado - 3;
break;
```

```
case "number":
resultado = resultado + "3";
break;
default:
resultado = "erro";
}
for (let i = 0; i < 3; i++) {
if (i == 1) {
var x = x * 2;
}
}
document.write(resultado + "<br>");
document.write(x);
</script>
```

---

a. Qual é a saída final exibida pelo código? Mostre o passo a passo completo do raciocínio

- Resultado:

203
20

- Passo a Passo:

O var é declarado assim: x = "10" (x é a string "10") e o let y = 5 (y é o número 5).

Chama-se a função calcular(10). Dentro da função existe var x local que recebe a + y. Como a é 10 (número) e y é 5 (número), a + y = 15 (número).

Dentro da função há if (x == "15"). O operador == faz coerção, então 15 == "15" é true. Então executa-se x = x + 5, ou seja 15 + 5 = 20 (número). A função retorna 20.

Após a função, resultado recebe 20 (número). O switch usa typeof resultado que é "number", então entra no case "number". Nesse case faz resultado = resultado + "3". Isso concatena número e string, produzindo a string "203".

No for, quando i == 1 executa-se var x = x * 2. A declaração var é no escopo global, portanto refere-se ao x global inicial

("10"). Na expressão x * 2 a string "10" é convertida para número, 10 * 2 = 20. O x global passa a ser o número 20.

E o final é o document.write - famoso print. Que imprime a saída que explicitei na primeira parte da resposta.

b. Explique onde ocorrem: hoisting, sombras de variáveis (variable shadowing), conversões
automáticas de tipos e coerção (==).

- Hoisting acontece com var, porque ele é puxado para o topo do código.

- Shadowing acontece quando existe um var x dentro da função, que esconde o x de fora.

- Coerção acontece quando o JS converte tipos automaticamente, como quando compara 15 com “15” usando ==, ou quando soma número com string e vira string.

c. No switch, por que o JavaScript cai naquele case específico? Justifique com base no tipo
real da variável naquele momento.

- O switch cai no case "number" porque typeof resultado é number naquele momento. A função calcular retorna 20, que é número.

d. No for, por que o valor de x muda mesmo ele tendo sido declarado antes do loop?
Explique usando regras de escopo var vs let.

- O valor de x muda porque var não tem escopo de bloco. O var dentro do for é o mesmo var global. Por isso, quando i é 1, ele faz x = x * 2 e muda o valor global de x.

e. o código corrigindo todos os possíveis problemas, substituindo var por let ou
const onde for adequado, e evitando conversões implícitas.

```
<script>
var x = "10";
let y = 5;

function calcular(a) {
var x = Number(a) + Number(y);
```

```
if (x === 15) {
    x = x + 5;
}
return x;

let resultado = calcular(10);

switch (typeof resultado) {
case "string":
resultado = Number(resultado) - 3;
break;

case "number":
    resultado = String(resultado) + "3";
    break;
default:
    resultado = "erro";

}

for (let i = 0; i < 3; i++) {
if (i === 1) {
x = Number(x) * 2; // conversão explícita e sem var novamente
}
}

document.write(resultado + "<br>");
document.write(x);
</script>

<script>
let x = "10";
const y = 5;

function calcular(a) {
let xLocal = Number(a) + Number(y);

  if (xLocal === 15) {
    xLocal = xLocal + 5;
  }

  return xLocal;

}
```

```
let resultado = calcular(10);

switch (typeof resultado) {
case "string":
resultado = Number(resultado) - 3;
break;

  case "number":
    resultado = String(resultado) + "3";
    break;

  default:
    resultado = "erro";

}

for (let i = 0; i < 3; i++) {
if (i === 1) {
x = Number(x) * 2;
}
}

document.write(resultado + "<br>");
document.write(x);
</script>
```