```
// ### Java Script - DESESTRUTURAÇÃO
//Capítulo: 03-04 (DESESTRUTURAÇÃO)

const obj = {
  id: 53,
  date: '2021-10-20',
  items: [
    {
      description: 'Celular',
      price: 1499.99,
      quantity: 1
    },
    {
      description: 'Mouse',
      price: 100.0,
      quantity: 2
    }
  ],
  client: {
    name: 'Maria Red',
    email: 'maria@gmail.com',
    active: true
  }
};

// COMO FUNCIONA A DESESTRUTURAÇÃO?

// Acima temos um objeto normal do JS chamado "obj"
// Se pegarmos este objeto e dermos um "console.log(obj)" ou digitarmos
//diretamente no console, iremos identificar
//este objeto criado com toda sua estruturação.
//{id, client, date, items}

//Então o que e esta Desestruturação?

//Aqui no JS é possível fazer uma atribuição conforme abaixo:
//Iremos criar uma constante "const"
//Em seguida abrir chaves e colocar APENAS ALGUNS MEMBROS DESTE OBJETO. Por exemplo:
"{ id, client }"
//Em seguida iremos fazer com que esta constante, QUE DEFINIMOS CONFORME ACIMA,
//ira receber uma referencia deste objeto "obj".
//Ficando assim:

const { id, client } = obj;

//Com isso estamos pegando duas variáveis "id" e "client", pegando
//diretamente a referência do objeto original "obj".

//Desta forma se executarmos o console.log(id) ou console.log(client),
//irá exibir no Console os valores "53" referente ao "id"
// e {name: 'Maria Red', email: 'maria@gmail.com', active: true},
// referente ao objeto do "client" que estava aninhado ao objeto "obj"

console.log(id);
console.log(client);
```

```
// OUTRO USO DA DESESTRUTURAÇÃO - COMO ARGUMENTO DE FUNÇÃO

//Suponha que tenhamos uma função "subtotal" recebendo o objeto "items"
// que se encontra aninhado dentro do objeto "obj".
// E aí o objetivo desta função seria retornar o subtotal deste item
//que é o valor do produto "price" multiplicado pela quantidade "quantity"
//Ficando assim:

function subtotal(item) {
  return item.price * item.quantity;
}

// Então se fizermos um console.log aplicado ao "obj.items" na posição "[1]",
conforme abaixo:

console.log(obj.items[1]);

//Irá exibir o seguinte objeto no console:
//{description: 'Mouse', price: 100, quantity: 2}

//Agora, vamos aplicar a função "subtotal" sobre este item.

console.log(subtotal(obj.items[1]));

//Irá exiber o valor: "200" no Console. Por que 200?
// Porque ele está considerando o Objeto Items[1] e capturando o preço "price" = 100
// e a quantidade "quantity" = 2, subtotalizando chegando ao valor de 200.

//Desta forma, temos a opção de ALTERNATIVAMENTE, em vez de ficar colocando o
//"nome_do_obeto.informação_que_precisamos" exemplo: "item.price", podemos
//optar pela DESESTRUTURAÇÃO.

// Para isto iremos substituir o arg0 da função, como no exemplo
//da função subtotal "function subtotal(item)", por { price, quantity } diretamente.
//Desta forma eliminamos a necessidade de ficar repetindo "item." toda hora.
// Ficando assim:

function subtotal({ price, quantity }) {
  return price * quantity;
}

console.log(subtotal(obj.items[1]));

// Irá exibir 200 no console.log da mesma forma.

// A DESESTRUTURAÇÃO COMO ARGUMENTO DE MÉTODO TAMBÉM É MUITO UTILIZADA.

// Agora para exercitar vamos montar um "console.log(total(obj))";
// Como se fosse um total de todos os "items"

//Iremos criar a função "function total()"
//Function total, que irá receber o objeto que seria a compra, que iremos chamar de
"order"
//e ela irá retornar o total da soma de todos os "items"
//Só que iremos fazer diferente.
//Esta função irá receber uma lista de "items" DESESTRUTURADOS.
```

```
117 //Repare que o objeto "obj" tem um campo que se chama "items".
118 //Desta forma já estamos DESESTRUTURANDO o obj e capturando diretamente
119 //todo o "items".
120
121 // Em seguida iremos iniciar uma variável "let soma = 0"
122 // E um for para percorrer todo o objeto "items"
123 // Dentro de for iremos atribuir a variável soma o valor dela mesmo
124 // mais a funçlão "subtotal" de "items" na posição [i]
125 // Retornando soma.
126 function total({ items }) {
127   let soma = 0;
128
129   for (let i = 0; i < items.length; i++) {
130     soma += subtotal(items[i]);
131   }
132   return soma;
133 }
134
135 console.log(total(obj));
136
137 //A execução deste console.log irá exibir o valor de 1699.99.
138 //Que é a soma do preço multiplicado pela quntidade de cada produto da lista
139 //neste caso: (Celular e Mouse).
140
141 // DESESTRUTURAÇÃO DE ARRAY (VETORES)
142
143 //Então podemos criar uma variável "const [item1, item2]"
144 //Atribuir a ela o "obj.items"
145
146 // Desta forma podemos atribuir o "obj.items" que é um vetor
147 // a uma estrutura DESESTRUTURADA dando nome para cada um desses elementos (item1 e
    item2)
148 // OBSERVAÇÂO: Se houvesse mais items neste vertor "items". Suponha que 4 items.
149 // Nesta desestruturação iríamos pegar apenas os dois primeiros items.
150 //Veja:
151
152 const[item1, item2] = obj.items;
153
154 console.log(item1); // Aqui será exibido no console o objeto items celular.
155 console.log(item2); // Aqui será exibido no console o objeto items mouse.
156
157
```