NÚMEROS CAPICÚAS

«Tu trabajo va a llenar gran parte de tu vida, la única manera de estar realmente satisfecho es hacer lo que creas que es un gran trabajo y la única manera de hacerlo es amar lo que haces. Si no lo has encontrado aún, sigue buscando. Como con todo lo que tiene que ver con el corazón, lo sabrás cuando lo hayas encontrado.»
Steve Jobs

"En matemáticas, la palabra capicúa se refiere a cualquier número que se lee igual de izquierda a derecha que de derecha a izquierda. Ejemplos: 161, 2992, 3003, 91019, 5005, 292, 2882, 2442, 9102019." Wikipedia

Crea un programa con una función recursiva que muestre todos los números/letras capicúas encontrados en un array. Se entiende por capicúa cuando tiene al menos 3 posiciones del array.
Por ejemplo con el array: [5, 4, 6, 5, 6, 4, 3, O, S, O]
mostrará:
656
46564
OSO

STARTER CODE:

```
<!DOCTYPE html>
<html>
<head>
<title>CAPICUA</title>
</head>
<body>
<p id="txt"></p>


<br><br>
Crea un programa con una función recursiva que muestre todos los
números/letras capicúas encontrados en un array. Se entiende por capicúa
cuando tiene al menos 3 posiciones del array.
Por ejemplo con el array: 5 4 6 5 6 4 3 O S O<br><br>
mostrará: <br>
656<br>
46564<br>
OSO
<p>
    <p>Pista:
```

```
        const animals = ['ant', 'bison', 'camel', 'duck', 'elephant'];

        console.log(animals.slice(2));
        // expected output: Array ["camel", "duck", "elephant"]

        console.log(animals.slice(2, 4));
        // expected output: Array ["camel", "duck"]
        console.log(animals.slice(2, 4).reverse());
        // expected output: Array ["duck","camel"]

    </p>

<script>

let arr =
["5","4","6","5","6","4","3","F","O","O","O","T","O","F","O","O","R","O","O","T","O"];
// ESCRIBE TU CÓDIGO AQUÍ

</script>
</body>
</html>
```

Corrección:

- Tu programa funciona y realiza todos los requisitos especificados: (3-5 puntos)
- Tu programa no realiza los requisitos especificados en el enunciado: (0-2 puntos)

Se valorará de tu programa:

- El programa realiza la funcionalidad requerida.
- Que no hay errores de consola.
- La eficiencia.
- La modularidad (alta cohesión y bajo acoplamiento).
- Que el código sea limpio, flexible, reutilizable y mantenible.
- Sigue los principios SOLID.
- Otros criterios.