Prueba de Nivel:

Algoritmia:

1. Demuestra que la siguiente cadena " **geeksforgeeks** " no es un palíndromo.

La palabra del enunciado **NO** puede leerse de derecha a izquierda. Las palabras capicúas o las frases que puede leerse tanto de izquierda a derecha como de derecha a izquierda, se denominan Palíndromos. Si copias y pegas el siguiente código en la consola de tu navegador, podrás comprobarlo.

En el caso de que ingrese una frase, será necesario eliminar los espacios en blanco.

```
<script>
function texto()
{
	var palabra = prompt ("Escribe una palabra o frase").toLowerCase();
	// eliminamos los espacios en blanco
	palabra = palabra.trim (“”)
	for (var i =0; i < palabra.length;  i++){
		if (palabra[i] != palabra [palabra.length-i-1]){
			return false;
		}
	}
	return true;
}
if(texto())
{
	console.log("El texto es palíndromo");
}else{
	console.log("El texto no es palíndromo")	}
</script>
```

CSS:

2. Diferencias entre flexbox y grid:

Cuando trabajamos con flexbox necesitamos pensar en términos de dos ejes, el eje principal y el eje cruzado. El eje principal está definido por la propiedad flex-direction, y el eje cruzado es perpendicular a este. Todo lo que hacemos con flexbox está referido a estos dos ejes.

En cambio un Grid o cuadrícula CSS se define usando el valor grid de la propiedad display, puede definir columnas y filas en su cuadrícula usando las propiedades grid-template-rows y grid-template-columns.

- Se podría pensar en flexbox como "unidimensional", y en grid como “bidimensional”.

- La cuadrícula se define principalmente en el elemento padre. En flexbox, la mayor parte del diseño (más allá de lo básico) ocurre en los elementos hijos.

- Grid es más resistente. Si bien la flexión de flexbox es a veces su fuerza, la forma en que se dimensiona un elemento flexible se vuelve bastante complicada.

- En ciertos navegadores, digamos que es más flexible usar flexbox, en vez de grid, debido al margen de compatibilidad y a la dinámica del diseño. Dependerá del caso.

HTML:

//1. Crea un div

```
div = document.createElement('div')
```

//2. Crea un SVG

```
document.createElementNS('http://www.w3.org/2000/svg','svg')

document.createElementNS('http://www.w3.org/2000/svg','div')
```

//3. Agrega una clase al div creado en el punto 1

```
div.classList.add('red')
```

//4. Agrega un estilo que cambie la propiedad color al punto 1

```
div.style.setProperty('color','blue')
```

//5. Agrega un atributo id al punto 1

```
div.setAttribute('id','maria')
```

//6. Crea una página en blanco y agrega al body el div con "hello world"

```
text = document.createTextNode('Hello World')
div.appendChild(text)
document.body.appendChild(div)
```

//7. Crea un nuevo div y agrégalo como hijo al punto 1 antes de "hello world"

```
child = document.createElement('div')
div.insertBefore(child,text)
```

//8. Dime los nodos que tiene el primer div

```
div.childNodes
```

//9. Dime los elementos que tiene el primer div

```
div.children
```

//10. Busca todos los div de la página

```
document.querySelectorAll('div')
```

//11. Busca los div que contienen la clase agregada al punto 3

```
document.querySelectorAll('.red')
document.getElementByClassName('red')
```

//12. Busca los div que tienen el id agregado en punto 5

```
document.querySelectorAll('#maria')
document.getElementById('maria')
```

//13. Deja el body sin elementos de tres formas diferentes

```
div.parentNode.removeChild(div)
div.remove()
```

```
document.body.textContent = ' '
```