Claro, aquí tienes una versión detallada de 10 ejercicios enfocados en **Estructuras de Control** (Condicionales y Bucles) en JavaScript, con una progresión gradual de dificultad.

Ejercicio 1: Determinar Par o Impar

Consigna:
Crea una función llamada `verificarParImpar` que reciba un argumento numérico entero: `numero`.
La función debe retornar el string `"Es par"` si el número es par, y el string `"Es impar"` si el número es impar.

Ejemplo:

- `verificarParImpar(4)` debería retornar `"Es par"`.
- `verificarParImpar(7)` debería retornar `"Es impar"`.

Ejercicio 2: Clasificación de Edad

Consigna:
Define una función llamada `clasificarEdad` que reciba un argumento numérico entero: `edad`.
La función debe retornar un string basado en el rango de edad:

- Si `edad` es menor de 13, retornar `"Niño/a"`.
- Si `edad` está entre 13 y 17 (inclusive), retornar `"Adolescente"`.
- Si `edad` es 18 o mayor, retornar `"Adulto/a"`.

Ejemplo:

- `clasificarEdad(10)` debería retornar `"Niño/a"`.
- `clasificarEdad(15)` debería retornar `"Adolescente"`.
- `clasificarEdad(25)` debería retornar `"Adulto/a"`.

Ejercicio 3: Día de la Semana

Consigna:
Escribe una función llamada `obtenerNombreDia` que reciba un argumento numérico entero: `numeroDia` (del 1 al 7, donde 1 es Lunes y 7 es Domingo).
La función debe retornar el nombre del día de la semana correspondiente. Si el número no está en el rango válido (1-7), debe retornar `"Día inválido"`.

Ejemplo:

- `obtenerNombreDia(1)` debería retornar `"Lunes"`.
- `obtenerNombreDia(7)` debería retornar `"Domingo"`.
- `obtenerNombreDia(0)` debería retornar `"Día inválido"`.
- `obtenerNombreDia(8)` debería retornar `"Día inválido"`.

Consideraciones:
Puedes usar una estructura `if/else if/else` o un `switch` para este ejercicio.

Ejercicio 4: Contar Hasta Cien

Consigna:
Implementa una función llamada `contarHastaCien` que no reciba ningún argumento.
La función debe usar un bucle `for` para imprimir en la consola cada número desde 1 hasta 100 (inclusive). No debe retornar ningún valor.

Consideraciones:
Deberás usar `console.log()` dentro del bucle para mostrar cada número.

Ejercicio 5: Suma de Números en un Rango

Consigna:
Crea una función llamada `sumarRango` que reciba dos argumentos numéricos enteros: `inicio` y `fin`.
La función debe retornar la suma de todos los números enteros desde `inicio` hasta `fin` (inclusive).

Ejemplo:

- `sumarRango(1, 5)` debería retornar `15` (1 + 2 + 3 + 4 + 5 = 15).
- `sumarRango(10, 10)` debería retornar `10`.

Consideraciones:
Puedes usar un bucle `for` o `while`.

Ejercicio 6: Imprimir Elementos de un Array

Consigna:
Define una función llamada `imprimirElementos` que reciba un argumento que sea un `array` de cualquier tipo de elementos.
La función debe usar un bucle (ya sea `for` o `forEach`) para imprimir cada elemento del array en la consola. No debe retornar ningún valor.

Ejemplo:

- Si el array es `["manzana", "banana", "cereza"]`, debería imprimir:

```
manzana
banana
cereza
```

Ejercicio 7: Contar Letras 'A' en un String

Consigna:
Escribe una función llamada `contarLetrasA` que reciba un argumento `texto` (string).
La función debe retornar la cantidad de veces que aparece la letra 'a' (mayúscula o minúscula) en el `texto`.

Ejemplo:

- `contarLetrasA("Banana")` debería retornar `3`.

- `contarLetrasA("JavaScript")` debería retornar `2` .
- `contarLetrasA("Hola Mundo")` debería retornar `1` .

Consideraciones:
Deberás iterar sobre el string y usar una condición para verificar cada caracter. Considera convertir el string a minúsculas para simplificar la comparación.

Ejercicio 8: Búsqueda de Elemento en un Array

Consigna:
Implementa una función llamada `buscarElemento` que reciba dos argumentos: `array` (un array de cualquier tipo) y `elementoBuscado` (el valor a buscar).
La función debe retornar `true` si `elementoBuscado` se encuentra en el `array` , y `false` en caso contrario.

Ejemplo:

- `buscarElemento([10, 20, 30], 20)` debería retornar `true` .
- `buscarElemento(["perro", "gato"], "pájaro")` debería retornar `false` .

Consideraciones:
Puedes usar un bucle `for` o `forEach` . La función debe detenerse y retornar `true` tan pronto como encuentre el elemento.

Ejercicio 9: Invertir un String

Consigna:
Crea una función llamada `invertirString` que reciba un argumento `cadena` (string).
La función debe retornar un nuevo string que sea la `cadena` original invertida.

Ejemplo:

- `invertirString("hola")` debería retornar `"aloh"` .
- `invertirString("JavaScript")` debería retornar `"tpircSavaJ"` .

Consideraciones:
Puedes iterar sobre el string de atrás hacia adelante o construir un nuevo string concatenando caracteres.

Ejercicio 10: Validar Contraseña (Complejidad Mínima)

Consigna:
Define una función llamada `validarContrasena` que reciba un argumento `contrasena` (string).
La función debe retornar `true` si la `contrasena` cumple con las siguientes condiciones, y `false` en caso contrario:

- Debe tener al menos 8 caracteres de longitud.
- Debe contener al menos una letra mayúscula.
- Debe contener al menos un número.

Ejemplo:

- `validarContrasena("abcD1234")` debería retornar `true` .

- `validarContrasena("abcd123")` debería retornar `false` (menos de 8 caracteres).
- `validarContrasena("ABCDEFGH")` debería retornar `false` (no tiene números).
- `validarContrasena("abcdefgh1")` debería retornar `false` (no tiene mayúsculas).

Consideraciones:
Necesitarás usar bucles para iterar sobre la contraseña y condicionales para verificar cada regla. Puedes usar métodos de string como `charCodeAt()` y rangos ASCII para verificar si un carácter es mayúscula o un número, o expresiones regulares básicas si las conoces (aunque se puede hacer sin ellas para este nivel).