PW EXAMEN- ejercicios varios

EJERCICIO: Contar Huecos en un Array

Crea una función llamada **contarHuecos** que reciba un array como parámetro y retorne el **número total de huecos** (posiciones vacías) que contiene.

```
function contarHuecos(arr)
{
  let cont=0;
  for(let i=0;i<arr.length;i++)
  {
    if(!(i in arr))
    {
      cont++;
    }
  }
  return cont;
}



const arr1 = [1, , , 4, , , , 5, 6, ,];
console.log(contarHuecos(arr1)); // 6

const arr2 = [1, 2, 3, 4, 5];
console.log(contarHuecos(arr2)); // 0

const arr3 = [, , , ,];
console.log(contarHuecos(arr3)); // 4

const arr4 = [10, , 20, , 30];
console.log(contarHuecos(arr4)); // 2
```

EJERCICIO: Eliminar Huecos de un Array

Crea una función llamada **eliminarHuecos** que reciba un array como parámetro y retorne un **nuevo array sin huecos**, conteniendo solo los elementos que realmente existen en el array original, manteniendo su orden.

```
function eliminarHuecos(arr){
  let res=[];
  for(let i=0;i<arr.length;i++)
  {
    if(i in arr)
    {
      res.push(arr[i]);
    }
  }
  return res;
}

const arr1 = [1, , , 4, , , , 5, 6, ,];
console.log(eliminarHuecos(arr1)); // [1, 4, 5, 6]

const arr2 = [10, 20, 30, 40, 50];
console.log(eliminarHuecos(arr2)); // [10, 20, 30, 40, 50]

const arr3 = [, , , ,];
console.log(eliminarHuecos(arr3)); // []

const arr4 = ['a', , 'b', undefined, 'c', , 'd'];
console.log(eliminarHuecos(arr4)); // ['a', 'b', undefined, 'c', 'd']

const arr5 = [, 100, , 200, , 300, ,];
console.log(eliminarHuecos(arr5)); // [100, 200, 300]
```

EJERCICIO: Análisis Completo de Arrays Dispersos

Crea una función llamada **analizarArrayDisperso** que reciba un array disperso y retorne un objeto con información detallada sobre su estructura.

El objeto retornado debe contener:

- **total**: longitud total del array

- **elementos**: cantidad de posiciones con valores (incluyendo `undefined` explícitos)
- **huecos**: cantidad de posiciones vacías
- **secuenciasDeHuecos**: array con las longitudes de cada secuencia consecutiva de huecos
- **posicionesConValor**: array con los índices que tienen valores
- **arrayLimpio**: nuevo array sin huecos

```
function analizarArrayDisperso(arr){
  let obj = {};
  let huecos = 0;
  let array_limpio = [];
  let array_secuencias_h = [];
  let array_posiciones_v = [];
  let inicio = null;
  let long = 0;

  for(let i = 0; i < arr.length; i++)
  {
   if(i in arr)  // Hay un valor
   {
    array_posiciones_v.push(i);
    array_limpio.push(arr[i]);

    if(inicio !== null)
    {
     array_secuencias_h.push(long);
     inicio = null;
     long = 0;
    }
   }
   else  // Es un hueco
   {
    huecos++;
    if(inicio === null) { inicio = i; }
    long++;
   }
  }
```

```
  if(inicio !== null)
  {
    array_secuencias_h.push(long);
  }

  obj = {
    total: arr.length,
    elementos: array_limpio.length,
    huecos: huecos,
    secuenciasDeHuecos: array_secuencias_h,
    posicionesConValor: array_posiciones_v,
    arrayLimpio: array_limpio
  };

  return obj;
}
const arr1 = [10, , , 20, , 30, 40, , , , 50];
console.log(analizarArrayDisperso(arr1));
/* Esperado:
{
  total: 11,
  elementos: 5,
  huecos: 6,
  secuenciasDeHuecos: [2, 1, 3],
  posicionesConValor: [0, 3, 5, 6, 10],
  arrayLimpio: [10, 20, 30, 40, 50]
}
*/

const arr2 = ['a', undefined, , 'b', , , null, , 'c'];
console.log(analizarArrayDisperso(arr2));
/* Esperado:
{
  total: 9,
  elementos: 5,
  huecos: 4,
  secuenciasDeHuecos: [1, 2, 1],
  posicionesConValor: [0, 1, 3, 6, 8],
```

```
  arrayLimpio: ['a', undefined, 'b', null, 'c']
}
*/

const arr3 = [1, 2, 3, 4, 5];
console.log(analizarArrayDisperso(arr3));
/* Esperado:
{
  total: 5,
  elementos: 5,
  huecos: 0,
  secuenciasDeHuecos: [],
  posicionesConValor: [0, 1, 2, 3, 4],
  arrayLimpio: [1, 2, 3, 4, 5]
}
*/

const arr4 = [, , , ,];
console.log(analizarArrayDisperso(arr4));
/* Esperado:
{
  total: 4,
  elementos: 0,
  huecos: 4,
  secuenciasDeHuecos: [4],
  posicionesConValor: [],
  arrayLimpio: []
}
*/
```

EJERCICIO: Sistema de Validación de Reservas Hoteleras

Una cadena hotelera necesita validar las reservas realizadas por sus clientes. Los datos están organizados en un array de objetos, donde cada objeto representa una reserva que cuenta con los siguientes atributos:

- **bookingId**: Identificador único de la reserva.

- **rooms**: Un array con las habitaciones reservadas. Cada habitación es representada por un objeto con:
  - **roomNumber**: número de la habitación (ejemplo: "101", "205", etc.).
  - **guests**: cantidad de huéspedes para esa habitación.
  - **maxCapacity**: capacidad máxima de la habitación.

El sistema debe verificar si la reserva es válida, es decir, que **todas** las habitaciones cumplan con la condición de que el número de huéspedes (`guests`) no supere la capacidad máxima (`maxCapacity`).

Escriba una función en JavaScript que reciba el array de reservas y devuelva un array con los identificadores (bookingId) de las reservas **válidas** (aquellas donde ninguna habitación excede su capacidad).

No está permitido el uso del bucle for clásico (`for (..;..;..)`), ni de métodos `map()`, `reduce()` o `filter()`, en caso de conocerlos.

```
function validarReservas(arr) {
  let validas = [];

  for (const reserva of arr) {
    let esValida = true;

    for (const habitacion of reserva.rooms) {
      if (habitacion.guests > habitacion.maxCapacity) {
        esValida = false;
        break; // Salir del bucle si encontramos una habitación inválida
      }
    }

    if (esValida) {
      validas.push(reserva.bookingId);
    }
  }

  return validas;
}

const bookings = [
```

```
  {
    bookingId: "BK001",
    rooms: [
      { roomNumber: "101", guests: 2, maxCapacity: 2 },
      { roomNumber: "102", guests: 3, maxCapacity: 4 }
    ]
  },
  {
    bookingId: "BK002",
    rooms: [
      { roomNumber: "201", guests: 5, maxCapacity: 4 },
      { roomNumber: "202", guests: 2, maxCapacity: 3 }
    ]
  },
  {
    bookingId: "BK003",
    rooms: [
      { roomNumber: "301", guests: 1, maxCapacity: 2 },
      { roomNumber: "302", guests: 2, maxCapacity: 2 },
      { roomNumber: "303", guests: 3, maxCapacity: 4 }
    ]
  },
  {
    bookingId: "BK004",
    rooms: [
      { roomNumber: "401", guests: 4, maxCapacity: 3 }
    ]
  }
];

console.log(validarReservas(bookings));
// Resultado: ["BK001", "BK003"]
```

Enunciados sacados de Claude y del pdf de profesor (Moodle)