Ejercicios de JavaScript Parte 3

1)
Dado un array de números, escribir una función que calcule la suma de todos los números del array.

2)
Dado un array de strings, escribir una función que devuelva la concatenación de todos los strings.

3)
Dado un array de números, escribir una función que devuelva un array con todos los números mayores a 10.

4)
Dado un array de números, escribir una función que devuelva la suma de todos los números del array que sean pares.

5)
Mediante el uso del método indexOf y splice, desarrollar una función que reciba un array de strings y un stringEliminar, la misma función nos devolverá un array sin el elemento que se eliminó. En caso de no tener el elemento a eliminar se deberá retornar null.

6)
Dado un array de nombres y un nombreDeLista, se deberá crear una función llamada crearLista que retorne un string con el siguiente formato.

Lista de (nombreDeLista):

1) **juan**
2) **pepe**
3) **maria**

7)
Dado un array de strings y un stringBuscado, elaborar una función que retorne la cantidad de veces que apareció en el array el stringBuscado.

8)
Crear un array vacio llamado gananciasMensuales.

Crear una función llamada vender, que reciba una cantidadVendida, un precio y un nombreDeProducto. La misma deberá agregar el precio total al array de gananciasMensuales.

9)
Crear un array vacio llamado gastosMensuales.

Crear una función llamada comprar, que reciba una cantidadComprada, un precio y un nombreDeProducto. La misma deberá agregar el total al array de gastosMensuales.

10)
Dado el array gastosMensuales y gananciasMensuales, desarrollar una función que nos retorne la gananciasMensuales. La misma se deduce de la diferencia entre la suma de valores del array de gastosMensuales y el de gananciasMensuales. (se puede usar la función hecha en el punto 1).

FUNCIONES FLECHA Y MÉTODOS BÁSICOS

11) Definí una función calcularAreaTriangulo que tome por parámetros la base y la altura de un triángulo y devuelva el área del mismo

// calcularAreaTriangulo(5, 6)
//resultado 15

12)Definí una función esElUltimoCaracter que reciba como argumentos una palabra y un caracter y nos indique si la palabra termina con el caracter.

//igualLongitud(‘estufa’, 'a')
//true

13)Definí una función esValida que tome por parámetro una contrasenia (string) y nos indique si tiene 8 caracteres o más.

//esValida('contraseniaMuySegura')
//true

14)Definí una función contarPalabras que reciba como argumento un string str y devuelva la cantidad de palabras que posee

//contarPalabras('javascript')
//1

15)Definí una función capitalizar que reciba como argumento un string str y devuelva el mismo string con la primera letra en mayúscula

//capitalizar('había una vez...')
//'Había una vez...'

16)Definí una función obtenerPrimeraOracion que tome por parámetro un string str y devuelva la primera oración de dicho string

//obtenerPrimeraOracion('A mí no me preguntes, sólo soy una oración')

//'A mí no me preguntes, sólo soy una oración'

17)Definí una función ocultarContrasenia que reciba como argumento una contrasenia de solo numeros y devuelva un string con dicha contraseña ocultada con astericos *, es decir, un string con la misma longitud donde todos sus caracteres son astericos

//ocultarContrasenia(123456)
//'****'**

18)Definí una función obtenerExtension que tome por parámetro un string archivo con el formato 'nombre.extension' y devuelva la extensión del archivo

//obtenerExtension('imagen.png')
//'png'

19)Definí una función esPuenteSeguro que tome por parámetro un string que represente a un puente que consista en caracteres numerales y espacios y nos indique si el puente está entero y es seguro atravesar

//esPuenteSeguro('### ##')
//false

ARRAYS

20)Definí una función obtenerMenor que tome por parámetro un array de números numeros y devuelva el menor de ellos

//obtenerMenor(5, 7, 99, 34, 54, 2, 12)
//2

21)Definí una función sumar que reciba como argumento un array de números numeros y devuelva la suma de ellos.

//sumar([5, 7, 10, 12, 24])
//58

22)Definí una función contiene que reciba como argumentos un número numero y un array de números numeros y devuelva si el número se encuentra en dicho array.

//contiene(103, [5, 7, 99, 3, 4, 54, 2, 12])
//false

23)Definí una función gano que reciba como argumento un array tragamonedas con 5 símbolos y nos indique si son iguales. Si el array tiene más de 5 símbolos, sólo debe comparar los 5 primeros.

//gano(['⭐ '⭐', '⭐', '💫', '✨']) //
//false

24)Definí una función separar que tome por parámetro un string con emojis de perros y gatos y devuelva un string con los perros agrupados por un lado y los gatos por otro.

//separar('🐶🐱🐶🐱🐱🐶🐶')
//'🐶🐶🐶🐶🐱🐱🐱'

25)Definí una función multiplicar que reciba como argumentos un número multiplicador y un array de números numeros, y que devuelva un array donde cada elemento es el resultado del elemento del primer array (en la misma posición) multiplicado por el número ingresado.

//multiplicar(2, [5, 7, 15, 22, 40])
//[10, 14, 30, 44, 80]

26)Definí una función filtrarPorLongitud que tome por parámetro un número longitud y un array de strings palabras y que devuelva un array con las palabras que tengan una cantidad de letras mayor a longitud.

//filtrarPorLongitudMayorA(4,['dia','remolacha','azul','sorpresa','te','verde',])
//['remolacha', 'sorpresa', 'verde']

27)Definí una función jugarPiedraPapelTijeras que reciba como argumentos dos strings jugadoraA y jugadoraB con los nombres de cada jugadora respectivamente, y dos arrays de strings jugadasA y jugadasB con jugadas de Piedra, Papel o Tijera, de la misma longitud. La función debe devolver un string con el nombre de la ganadora o Empate en caso de que no haya ninguna. Para eso, debe comparar las mismas posiciones de cada array de jugadas, y sumar puntos a la jugadora correspondiente

const jugadasA = ['piedra', 'piedra', 'tijera']
const jugadasB = ['papel', 'tijera', 'tijera']

<u>COMBINADOS</u>

28)Definí una función esPalindromo que tome una palabra y devuelva si dicha palabra es palíndroma, es decir, si puede leerse de igual manera de izquierda a derecha que de derecha a izquierda .

//esPalindromo('mama')
//false

29)Definí una función repetirLetras que reciba como argumento un string palabra y un número entero cantidad, y devuelva una string donde cada letra de palabra esté repetida cantidad de veces.

//repetirLetras('ah!', 5)
//aaaaahhhhh!!!!!'

30)Definí una función removerDuplicados que tome por parámetros un array array y que devuelva un array con los mismos valores de array pero sin valores duplicados.

//removerDuplicados([1, 1, 1])
//[1]

<u>METODOS AVANZADOS</u>

31) definí la función dobles que tome una lista de numeros y devuelva un nuevo array con cada valor multiplicado por dos

//dobles(numeros)
//[6, 14, 26, 198]

32)definí una función longitudes que tome una lista de frases y devuelva un nuevo array que contenga la longitud de cada frase.

//longitudes(frases)
//[17, 18, 16]

33)definí la función posiciones que tome una lista de canciones de una playlist y devuelva un nuevo array con los números de las posiciones de cada canción.

//const playlist = ['Everlong', 'The Pretender', 'Learn to Fly'];

posiciones(playlist)
['0 - Everlong', '1 - The Pretender', '2 - Learn to Fly']

34)definí la función losMasCaros que tome una lista de costos que representan costos de diferentes productos y devuelva un nuevo array con los precios más caros (mayores a 50)

//const costos = [39, 53, 17, 99, 7, 9, 6, 68, 54, 97, 27, 90, 92, 75, 26, 86, 22, 42, 20, 14]

// losMasCaros(costos)
[53, 99, 68, 54, 97, 90, 92, 75, 86]

35)definí la función soloStrings que tome una lista de mix con varios elementos de distintos tipos de datos y devuelva un nuevo array que contenga sólo los datos de tipo string.

```
const mix = [ 'Ut vero.',2 , function () { console.log('hola mundo!') }, 56, 'Diam rebum nonumy et.', true, false,'Kasd stet.', 'Sit et dolor.', null, null, [ 1, 2, 3], 'Dolore.'];

//soloStrings(mix)
//['Ut vero.', 'Diam rebum nonumy et.', 'Kasd stet.', 'Sit et dolor.', 'Dolore.' ]
```

36)

Necesitamos crear un contador . Para esto necesitaremos

- **un span que muestre un número (que comience en 0);**
- **6 botones con los siguientes valores: : -1, +1, -5, +5, -10, +10;**
- **al clickear un botón se tiene que actualizar el valor del span sumando o restando la cantidad correspondiente.**

37)
Vamos a crear una página para jugar a resolver una pregunta . Se espera que la misma:

- **tenga dos span el primero con una pregunta, o adivinanza, el segundo inicialmente estará vacío;**
- **tenga tres botones con posibles respuestas;**
- **al clickear la respuesta correcta, el botón debe ponerse de color verde, y el texto del segundo span debe actualizarse mostrando ¡Respuesta correcta!;**
- **si se clickea una respuesta incorrecta, se debe mostrar el botón con la respuesta correcta con un color de fondo verde y los otros dos con un color de fondo rojo, y el texto del segundo span debe actualizarse mostrando ¡Respuesta equivocada!.**

38)Vamos a crear una página que inicialmente cuente con 5 imágenes:

- **las cuatro primeras imágenes tendrán 100px de alto;**
- **la última deberá tener 500pxde alto.**

Al clickear una imagen de 100px se actualiza a 500px y la que anteriormente tenía 500px se actualiza a 100px.

39)
Necesitamos darle colores aleatorios al body apretando la barra espaciadora. Para eso modificaremos su color de fondo utilizando el formato rgb(0,0,0) donde los valores r, g y b deben generarse aleatoriamente con valores entre 1 y 255.

40)Queremos obtener el nombre, año de lanzamiento y banda de un disco , para mostrar el mensaje "El disco [NOMBRE DISCO] de la banda [NOMBRE DE LA BANDA] se lanzó en el año [AÑO DE LANZAMIENTO DEL DISCO]".

```
let disco = {
  id: 1,
  nombre: 'Wasting Light',
  anioLanzamiento: 2011,
  cantidadDeTemas: 12,
  banda: {
    nombre: 'Foo Fighters',
    anioFormacion: 1994
  }
};

//infoDelDisco(disco);
//"El disco Wasting Light de la banda Foo Fighters se lanzó en el año 2011"
```

41)A partir de un array de bandas queremos saber si están activas o no

```
let bandas = [
  { id: 1, nombre: "Nirvana", fundacion: 1987, activa: false },
  { id: 2, nombre: "Foo Fighters", fundacion: 1994, activa: true },
  { id: 3, nombre: "Led Zeppelin", fundacion: 1967, activa: false },
  { id: 3, nombre: "Queens of the Stone Age", fundacion: 1997, activa: true },
  { id: 3, nombre: "Pearl Jam", fundacion: 1990, activa: true },
];

//estanActivas(bandas)
//
`
Nirvana no está activa
Foo Fighters está activa desde el año 1994
Led Zeppelin no está activa
Queens of the Stone Age está activa desde el año 1997
Pearl Jam está activa desde el año 1990
`
```

42)Para modelar bandas contamos con objetos con su nombre, año de lanzamiento, si sigue en actividad, una foto, el listado de integrantes y una lista de sus discos

```
let ledZeppelin = {
  nombre: "Led Zeppelin",
  anio: 1968,
  activa: false,
  miniatura:
    "http://townsquare.media/site/295/files/2014/10/Led-Zeppelin1.jpg?w=980&q=75",
  integrantes: ["Jimmy Page", "Robert Plant", "John Paul Jones", "John Bonham"],
  discos: [
    {
      nombre: "Led Zeppelin",
      anio: 1969,
      canciones: ["Good Times, Bad Times", "Communication Breakdown"],
      duracion: 2691,
    },
    {
      nombre: "Led Zeppelin II",
      anio: 1969,
      canciones: ["Whole Lotta Love", "Moby Dick", "Ramble On"],
      duracion: 2502,
    },
    {
      nombre: "Led Zeppelin III",
      anio: 1970,
      canciones: ["Immigrant Song"],
      duracion: 2489,
    },
    {
      nombre: "Led Zeppelin IV",
      anio: 1971,
      canciones: ["Rock and Roll", "Stairway to Heaven", "Four Sticks"],
      duracion: 2559,
    },
  ],
};

//informacionDeLaBanda(ledZeppelin)
//"Led Zeppelin se fundó en el año 1968. Tiene 4 integrantes: Jimmy Page, Robert Plant, John Paul Jones, John Bonham. Tiene en total 4 discos. Tiene en total 9 canciones entre todos los discos. En promedio, cada canción dura 1137.888888888889 segundos."
```

43)Queremos visualizar fácilmente las propiedades y valores de nuestros productos con el siguiente formato: "producto[PROPIEDAD] -> VALOR". Para ello definiremos una función

```
let producto = {
  id: "ADA-020",
  titulo: "Gubergren sed est amet voluptua",
  precio: 123.75,
  imagen:

"https://i.kinja-img.com/gawker-media/image/upload/s--53mPze4a--/c_scale,f_auto,fl_progressive,q_80,w_800/1315358249455433031.jpg",
  condicion: "nuevo",
  envioGratis: true,
  ubicacion: "Capital Federal",
};

//obtenerPropiedadesYValores(producto)
//
`
producto['id'] -> ADA-020
producto['titulo'] -> Gubergren sed est amet voluptua
producto['precio'] -> 123.75
producto['imagen'] ->
https://i.kinja-img.com/gawker-media/image/upload/s--53mPze4a--/c_scale,f_auto,fl_progressive,q_80,w_800/1315358249455433031.jpg
producto['condicion'] -> nuevo
producto['envioGratis'] -> true
producto['ubicacion'] -> Capital Federal
`
```

44)Contamos con perfiles que tienen nombreDeCuenta y contrasenia y pueden tener o no el campo email

```
let perfilSinEmail = {
  nombreDeCuenta: "ada_lovelace",
  contrasenia: "1234567890!"
};

let perfilConEmail = {
  nombreDeCuenta: "ellie_arroway",
  contrasenia: "vegaeterna",
  email: "ellie@argos.org"
};

//tieneEmail(perfilConEmail)
//'El usuario tiene la propiedad email'
```

45)enemos los datos de distintas personas que quieren crear un perfil dentro de Gmail guardados de la siguiente forma:

```
let perfil1 = {
  nombre: 'Grace',
  apellido: 'Hopper',
  email: 'grace.hopper@gmail.com',
  password: '123456'
};

let perfil2 = {
  nombre: 'Ada',
  apellido: 'Lovelace',
  email: 'ada.lovelace@gmail.com',
  password: '**178!Ada--'
};

let perfil3 = {
  nombre: 'Hedy',
  apellido: 'Lamarr',
  email: 'hlamarr@gmail.com',
  password: '1234'
};
```

Nos pidieron implementar una función, llamada validarPassword, que reciba un perfil y valide la contraseña.

- **la función nos tiene que retornar un objeto con dos propiedades: verificada (un booleano) y mensaje (un string que contiene el mensaje de la validación que falló, o vacío si salió todo bien);**
- **de la contraseña tenemos que validar:**
    - **que la longitud sea mayor o igual a 6 (si es menor, retornar el mensaje "Contraseña con menos de 6 caracteres");**
    - **que la contraseña no sea una de: "123456", "password", "111111", "qwerty" (si coincide con alguna de esas contraseñas, retornar el mensaje "Contraseña muy insegura")**

```
//validarPassword(perfil3)
//{ verificada: false, mensaje: 'Contraseña con menos de 6 caracteres' }
```

46)Queremos crear un html a partir de un objeto de JavaScript, más especificamente vamos a crear una página que nos permita visualizar discos de Spotify . Los discos tienen las propiedades: id (string), nombre (string), anio (número), genero (array de strings), banda (string), portada (string) e info (string).

```
let albumDeEjemplo = {
  id: 'nirv1234',
  nombre: 'With The Lights Out',
  anio: 2004,
  genero: 'Grunge',
  banda: 'Nirvana',
  portada:
'https://muzikalia.com/wp-content/uploads/2005/03/nirvana__with_the_lights_out.jpg',
  info: 'https://en.wikipedia.org/wiki/With_the_Lights_Out'
};
```

Para eso definiremos una función render que reciba un disco como argumento y genere un HTML de la siguiente forma:

```
//render(albumDeEjemplo)
//
`
<div class="card m-5" id="nirv1234">
  <img class="card-img-top"
src="https://muzikalia.com/wp-content/uploads/2005/03/nirvana__with_the_lights_out.
jpg" alt="Nirvana - With The Lights Out" />
  <div class="card-body">
    <h5 class="card-title">Nirvana</h5>
    <p class="card-text">With The Lights Out (2004) | Grunge</p>
    <a href="https://en.wikipedia.org/wiki/With_the_Lights_Out" class="btn
btn-primary">https://en.wikipedia.org/wiki/With_the_Lights_Out</a>
  </div>
</div>
`
```

47)Necesitamos definir una función que dado un objeto con productos y precios, y la cantidad de dinero disponible, devuelva un objeto con dichos productos, teniendo como valor true si puede comprarlo o false sino.

```
let productos = { cookies: 60, chocolate: 110, soda: 120 };
let dinero = 115;

//comprarProductos(dinero, productos)
//{ cookies: true, chocolate: true, soda: false }
```

48)Necesitamos una función que dado un string devuelva un objeto con la cantidad de letras, espacios y números que contiene. Cualquier cosa que no sea un número o un espacio cuenta como una letra.

```
//obtenerInfoString("H3ll0 Wor1d")
//{letras:  7, digitos: 3, espacios: 1}
```

49)Necesitamos una función que dada una persona y una búsqueda de empleo, devuelva si la persona se ajusta a dicha búsqueda.

```
let persona = {
  experiencia: 4,
  lenguajes: ["JavaScript", "HTML","CSS"],
  locacion: "Buenos Aires",
  remuneracion: 35000
}

let busqueda = {
  experiencia: 1,
  lenguajes: ["JavaScript", "HTML"],
  locacion: ["Buenos Aires"],
  remuneracion: 40000
}
```

La persona se adecua si:

- **cuenta con los años de experiencia necesarios para la búsqueda;**
- **los lenguajes que conoce son los que contiene la búsqueda (puede conocer más, pero no afecta en nada);**
- **su locación está incluida dentro de las locaciones posibles de la búsqueda;**
- **su remuneración (pretendida) es igual o menor a la de la búsqueda.**

```
//seAdecua(persona, busqueda)
//true
```