Ejercicio: 001__001

En España el documento nacional de identidad se conoce como DNI.
Cada DNI tiene un número de identificación único.
Este número está compuesto por un número de 8 cifras, acompañado de una letra de seguridad.
La letra asegura la integridad del DNI, ya que se calcula en base a los números.

Ejemplos de DNI:

12312312K
78163312C
12345678Z
34667892S
92234488A

El cálculo de la letra se realiza de la siguiente forma:

Se toma el número y se divide entre 23.
Según el resto de la división entre 23 miramos la letra correspondiente en la siguiente tabla:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
T	R	W	A	G	M	Y	F	P	D	X	B	N	J	Z	S	Q	V	H	L	C	K	E

Ejercicio:

Realiza una función que reciba el número de un DNI y devuelva la letra correspondiente.
Puedes realizar pruebas con los números de DNI de los ejemplos anteriores.

Ejercicio: 001__002

La función desarrollada en el ejercicio anterior cumple con su cometido.
Pero solo funciona correctamente en los casos óptimos.

¿Qué pasaría si le indicamos un número de 3 letras?
¿Qué pasaría si en vez de un número le enviamos un string?

Ejercicio:

Reutiliza la función del ejercicio anterior e introduce las siguientes mejoras:

1) Si introducimos un valor no numérico deberá mostrar el mensaje: "Debes introducir un valor numérico."
2) Si introducimos un valor numérico que no tenga 8 cifras deberá mostrar: "Debes introducir un número de 8 cifras."
3) Si introducimos un valor numérico negativo deberá mostrar: "Debes introducir un valor positivo."

BONUS:

Si el número del DNI tiene menos de 8 dígitos, debemos añadirle ceros por la izquierda hasta completarlo. La función devolverá el DNI resultante

Ejercicio: 001__003

Realiza una función que reciba un array de strings y devuelva la longitud del string más largo

Por ejemplo, dado el siguiente array:

var array = ["Hola", "Frase corta", "Frase normalita", "Frase muy muy larga"];

Deberá devolver: 19

Bonus:

1) Comprobar que todos los valores son de tipo string
2) Devolver un objeto que contenga el string y el número de caracteres
ejemplo:

```
{
        longitud: 19,
        string: "Frase muy muy larga"
}
```

Ejercicio: 001__004

Vamos a complicar el ejercicio anterior:
Ahora cada vez que calculemos la longitud del string más largo, almacenaremos el resultado en un array de resultados.

Una vez ejecutados varios cálculos de longitud (4 en el ejemplo), vamos a calcular la media de los resultados.

Código de ejemplo para empezar:

```
var resultados = [];

var arrayDeTest1 = ["Richie", "Joanie", "Greg", "Marcia", "Bobby"];
var arrayDeTest2 = ["Blanka", "Zangief", "Chun Li", "Guile"];
var arrayDeTest3 = ["Red", "Blue", "Green"];
var arrayDeTest4 = ["Hola", "Frase corta", "Frase normalita", "Frase muy muy larga"];


var calculoLongitudMasLargo(arrayDeTest1);
var calculoLongitudMasLargo(arrayDeTest2);
var calculoLongitudMasLargo(arrayDeTest3);
var calculoLongitudMasLargo(arrayDeTest4);

console.log(resultados);
```

Ejercicio: 001__005

Realiza una función que reciba un string y devuelva un objeto contando el número de apariciones de cada letra en el string.
Almacena y devuelve el resultado en un objeto.

Ejemplo de resultado:

```
resultado = {
        a: 3,
        d: 7,
        f: 4
}
```

```
// AYUDA, para partir el string puedes hacer uso de:
var arrayDeCaracteres = nombreDeString.split("");
```

```
// Tests para comprobar tu función

resultadoContador = contadorDeCaracteres("abbabcbdbabdbdbabababcbcbab");
console.log( resultadoContador['a'] === 7);
console.log( resultadoContador.b === 14);
console.log( resultadoContador['c'] === 3);

resultadoContador = contadorDeCaracteres("xyyyxyxyxzyxyzyxyxyasdfz");
console.log( resultadoContador.x === 7 );
console.log( resultadoContador['y'] === 10 );
console.log( resultadoContador.z === 3 );
console.log( resultadoContador['a'] === 1 );
console.log( resultadoContador['s'] === 1 );
console.log( resultadoContador.d === 1 );
console.log( resultadoContador['f'] === 1 );
```

Ejercicio: 001__006

Completa las funciones siguientes para tener un conjunto de operaciones que traten nuestros arrays realizando diversas transformaciones sobre ellos.

Nota: puede que alguna de las siguientes funciones te sea de ayuda:

isFinite(), splice(), unshift(), push(), join(), sort(), Math.floor()

Puedes encontrar más en:

https://developer.mozilla.org/en-US/docs/Web/JavaScript
y
http://www.w3schools.com/js/default.asp

```
function vaciarPapelera(array) {
    // Esta función debe recibir un array y devolverlo habiéndole
quitado los elementos que sean un asterisco (*)
    // Por ejeplo:
    // vaciarPapelera(['a',1,'*',5])
    // debe devolver:
    // ['a',1,5]
}

function agruparElementos(array) {
    // Esta función debbe recibir un array con números y letras y
devolverlo habiendo agrupado los elementos
    // En primer lugar se deben encontrar números, depués letras. El
orden dentro de cada grupo no importa.
    // Por ejemplo:
    // agruparElementos(['B', 'a', 4 , 23, 'J'])
    // debe devolver:
    // [23, 4, 'B', 'a', 'J']
}

function ponerBonitasLasLetras(array) {
    // Esta función debe recinbir un array de números y letras y
devolverlo con las letras vocales en mayúsculas
    // y las consonantes en minúsculas. Los números no deben ser
tratados.
    // Por ejemplo:
    // ponerBonitasLasLetras([1,5,7,'a','J',p,'E'])
    // debe devolver:
    // [1,5,7,'A','j',p,'E']
}
```

```
function ponerBonitosLosNumeros(array) {
    // Esta función debe recibbir un array de números y letras, y
devolverlo con los números "bonitos".
    // Las letras no deben cambiar.
    // Para poner bonito número debemos sumar todas sus cifras.
    // en caso de que el número resultante tenga más de una cifra, se
petirá el proceso con este.
    // 123 se convertirá en 6 porque 1+2+3 = 6
    // 9 se convertirá en 9
    // 9956 se convertirá en 2 porque 9+9+5+6 = 29, 2+9 = 11 y 1+1 = 2
    // 793 se convertirá en 1 porque 7+9+3 = 19, 1+9 = 10 y 1+0 = 1
    // Este proceso debemos realizarlo sobre un array de elementos y
aplicarlo solo a los números.
}

function arrayToString(array) {
    //Esta función debe recibir un array y devolver un string con
todos sus elementos
    //Ejemplo: arrayToString([1, 4, 5, 5, 'A', 'b', 'E', 'j']) dee
devolver "1455AbEj"
}
```

Ejercicio: 001__007

Realiza una función que reciba un array de números y devuelva cada par de posiciones que al sumarlas devuelvan cero.

Por ejemplo:

Posiciones: 0 1 2 3 4 5 6 7
  var array = [2, -5, 10, 5, 4, -10, 0, -5];

Ejemplo de función

```
var buscarParejas = function(array) {

}
```

Tests

```
var miArray = [2, -5, 10, 5, 4, -10, 0, -5];
console.log(buscarParejas(miArray));
```

Debe imprimir ["1,3" , "2,5" , "3,7", "6,6"]

Ejercicio: 001__008

Escribe una función que reciba un string de números separados por dos puntos, cree un array a partir del string y devuelva la media de todos los valores

Tests:

var stringDeNumeros = '80:70:90:100';
La función debe devolver 85

Bonus:

Misma funcionalidad, pero eliminando los repetidos

var stringDeNumeros = '80:70:90:100:100:100:100';
también deberá devolver 85

Ejercicio: 001__009

Realiza las funciones siguientes

```
// Esta función recibe un string y devuelve el string inverso
// Por ejemplo: para el string "Hola clase!" debe devolver "!esalc
aloH"
function stringInverso(string) {

}

// Esta función debe recibir un string y devolver el mismo string sin
espacios
function eliminarEspacios(string) {
        // Con expresión regular
        cadena = cadena.replace(/\s/g, '');
}

// Esta función debe recinir un string y devolverlo con todas sus
letras mayúsculas
function ponerTodasLasLetrasMayusculas(string){

}

// Esta función debe recibir un string y decir si es un palíndromo
(true / false)
// Un palíndromo es una frase que se lee igual al derecho que al revés
// Haz uso de las tres funciones anteriores
function esPalindromo(string) {

}
```

Ejemplos de palíndromos:

Arde ya la yedra
Ana lava lana
Anita lava la tina

Ejercicio: 001__010

Realiza la modelización de un Zoológico

El zoológico deberá tener un nombre, una ubicación, un aforo máximo, un horario... ¡y todo lo que se te pueda ocurrir!

El zoológico deberá tener varias áreas:

- Reptiles
- Aves
- Mamíferos
- Peces

Cada área deberá tener distintos recintos para los animales, por ejemplo:

- Reptiles
    - Serpientes
    - Lagartos
    - … etcétera
- Aves
    - Aves pequeñas
    - Aves tropicales
    - ... etcétera

Cada recinto debe tener un nombre, una capacidad máxima de animales, aforo máximo de personas y un conjunto de animales.

Modeliza el zoológico lo más completo que puedas.

Puedes continuar a partir del código siguiente:

```
var zoo = {
        nombre: "El último zoológico",
        ubicacion: {},
        areas: [],
        aforo: 120
        // COMPLETAR
};

zoo.ubicacion = {
        direccion: "Calle de los animales 5",
        ciudad: "París",
        pais: "Francia",
        // COMPLETAR
}

var area1 = {
        nombre: "Reptiles",
        aforoMaximoZona: 30,
        recintos: [], // son como jaulas
        // COMPLETAR
}

function crearArea(nombreRecibido, aforoRecibido){
        return {
                nombre: nombreRecibido,
                aforoMaximoZona: aforoRecibido,
                recintos: []
        }
}

var area1 = crearArea("Reptiles", 10);
var area2 = crearArea("Aves", 10);
var area3 = crearArea("Animales acuáticos", 10);
zoologico.areas.push(area1, area2, area3);


zoo.areas.push(area1);
```

Ejercicio: 001__011

Haciendo uso del zoo que definimos en el ejercicio anterior,
vamos a añadirle funcionalidad:

1) Haz una función que añada un visitante nuevo:
    Si el zoo está lleno no podrá entrar.
    Para entrar deberá pagar la entrada que dependerá de:
        Niños <14 años: gratis
        Personas mayores >65 gratis
        Resto: 5$
        Estudiantes: 3$
    Este importe deberá ser restado del dinero del visitante y añadido a la caja del zoo

    El visitante irá a un área y una sección aleatoria,
    si esta está llena, deberá buscar otro lugar

2) Crea una función que se llame ejecutarCiclo() que simule el paso de 1 hora en el zoo, deberá:
    - Añadir visitantes al parque y también los retire del parque
    - Deberá quedar reflejado que ha pasado un ciclo en el importe de las personas (tendrán menos dinero) y en la caja del parque (habrá ganado dinero)

Nota: (El ciclo simula ser una hora del parque, pero lo ejecutamos cada 3seg)

3) Crea una funcionalidad que simule el paso de un ciclo en un animal:
    - Su salud se verá afectada disminuyendo o aumentando 10 (de forma aleatoria).
    - Si la salud del animal desciende por debajo de 50, este deberá ir a la enfermería.
    - También el animal tendrá más hambre cada hora que pase (+10) cuando llegue a 100 deberá ser alimentado y pasará a tener hambre 0.

4) Asocia la funcionalidad anterior a la función de ejecutarCiclo() de manera que los animales vayan variando su salud y su hambre.
De vez en cuando algunos animales deberán ir a la enfermería (salud menor de 50) donde recuperarán 10 de salud hasta llegar a 100.
Al llegar a 100 deberán volver a su recinto.

Ejercicio: 001__012

Partiendo del ejercicio anterior:
Partiendo del ejercicio anterior:

1) Crear función cerrar zoo:
    - Parar el intervalo
    - Sacar a todas las personas del zoo

2) Crear función modificarSaludAleatoria(animal) que de manera aleatoria sume o reste salud a un animal: aleatorio entre -20 y +20 (maximo de salud es 100).

3) En cada ciclo ejecutar la funcion de modificarSaludAleatoria() para todos los animales, si alguno baja de 50 de salud, deberá ir a la enfermería

4) En la enfermería en cada ciclo los animales recuperan 10 de salud (no se aplica modificarSaludAleatoria()).

5) Si el animal llega a 100 de salud deberá volver a su área (debemos saber a qué area pertenecía)

6) Crear función addHambre() que en cada ciclo sume 10 al hambre de un animal. Cuando llegue a 100 el animal estará muy hambriento y deberá ser alimentado. Al alimentar un animal su hambre pasa a 0 y al zoo le cuesta 1000$

7) Si el zoo no tiene dinero para alimentar a los animales, no podrá hacerlo. Cuando un animal tenga más de 150 de hambre, se comerá a un visitante. El zoo se quedará con su cartera.

8) En cada ciclo los visitantes deberán cambiar al siguiente recinto. Cuando hayan visitado todos abandonarán el parque.

1) Crear función cerrar zoo:
    - Parar el intervalo
    - Sacar a todas las personas del zoo

2) Crear función modificarSaludAleatoria(animal) que de manera aleatoria sume o reste salud a un animal: aleatorio entre -20 y +20 (maximo de salud es 100).

3) En cada ciclo ejecutar la funcion de modificarSaludAleatoria() para todos los animales, si alguno baja de 50 de salud, deberá ir a la enfermería

4) En la enfermería en cada ciclo los animales recuperan 10 de salud (no se aplica modificarSaludAleatoria()).

5) Si el animal llega a 100 de salud deberá volver a su área (debemos saber a qué area pertenecía)

6) Crear función addHambre() que en cada ciclo sume 10 al hambre de un animal. Cuando llegue a 100 el animal estará muy hambriento y deberá ser alimentado. Al alimentar un animal su hambre pasa a 0 y al zoo le cuesta 1000$

7) Si el zoo no tiene dinero para alimentar a los animales, no podrá hacerlo. Cuando un animal tenga más de 150 de hambre, se comerá a un visitante. El zoo se quedará con su cartera.

8) En cada ciclo los visitantes deberán cambiar al siguiente recinto. Cuando hayan visitado todos abandonarán el parque.