Ejercicios Unidad 2: Arrays y objetos definidos por el usuario.

Objetos y Clases

EJERCICIO: u3e01_cuenta

Escribe un programa que cree un objeto "**cuenta**" a partir de un literal con las siguientes propiedades:

- Una propiedad **titular** con el valor "Alex".
- Una propiedad **saldo**, teniendo como valor inicial 0.

Crea funciones que nos permitan realizar las siguientes operaciones:

- **ingresar()** que permita añadir dinero a la cuenta, pasando la cantidad como parámetro.
- **extraer()** que permita retirar la cantidad pasada como parámetro.
- **informar()** que retorne la información del estado de la cuenta.

A continuación, añade al objeto las propiedades **saldoMáximo** con valor inicial 1000 y **fechaApertura** con valor inicial a la fecha actual. Por último, elimina la propiedad **saldoMáximo.**

Todas las operaciones que se realicen se irán mostrando en la página.

EJERCICIO: u3e02_clientes

Realiza de nuevo el ejercicio **u2e06_clientes** pero en este caso con un array de objetos creados a partir de un literal de objetos o bien, con objetos anidados, tú eliges. Se trata de que utilices los métodos que podemos aplicar al manejo de objetos. Aquí tienes el enunciado de nuevo:

Dispones del siguiente archivo de texto:

```
Cliente;Localidad;Cuota
Laura;Santander;50
Álvaro;Castro;50
Igor;Castro;60
Ivan;Santander;40
Mónica;Zamora;30
Javi;Bilbao;30
David;Bilbao;50
José Luis;Bilbao;60
```

A partir del mismo, el usuario podrá elegir del menú:

1. Todos: se mostrará una tabla con los valores que están en la variable anterior.
2. Clientes de una localidad: se pedirá una localidad al usuario y se mostrarán en una tabla los nombres y cuotas de las personas que viven en esa localidad.
3. Clientes que tengan una cuota mayor que un valor pedido al usuario: se pedirá la cuota y se mostrarán en una tabla los nombres de clientes, localidad y cuotas de aquellos que tienen una cuota superior al valor introducido por el usuario.

EJERCICIO: u3e03_discos

Crea una clase **Disco** que almacene la siguiente información:

- Nombre del disco.
- Grupo de música o cantante.
- Año de publicación.
- Tipo de música (podrá ser “rock”, “pop”, “punk” o “indie”);
- Localización: almacenará un número de estantería.
- Prestado: almacenará un valor booleano. Por defecto será false.

Además, tendrá los siguientes métodos:

- Un constructor que se puede llamar con o sin parámetros:
  - Sin parámetros: las 4 primeras propiedades serán cadenas vacías, la localización será 0 por defecto y prestado estará a false.
  - Con parámetros: se pasarán solo las cinco primeras propiedades; la propiedad prestado será false.
- Un método que permitirá cambiar el número de estantería en la localización.
- Un método que permitirá cambiar la propiedad prestado.
- Un método que muestre toda la información de un disco.

El programa hará lo siguiente:

- Creará un objeto de tipo disco, para lo cual te pide los parámetros.
- Presente un menú que permita:
  - Mostrar la información del disco que se ha añadido
  - Cambiar el disco de estantería
  - Prestar un disco
  - Devolver un disco
  - Terminar

EJERCICIO: u3e04_vector

Crea una clase **Vector** que representa un vector. Al crearlo, se pasarán como parámetros dos valores numéricos que serán la **x** (distancia al punto 0,0 en el eje de las x), y la **y** ((distancia al punto 0,0 en el eje de las y).

La clase tendrá definidos los métodos **get** y **set** correspondientes a las propiedades x e y.

Esta clase tendrá al menos dos métodos más: **sumar** y **restar**, que toman otro vector como parámetro, y devuelve un nuevo vector resultado de la suma o diferencia de los valores x e y de los dos vectores (el this y el parámetro). Estos dos métodos, deben ser estáticos.

Además, tendrá una propiedad **longitud**, que devuelve la longitud del vector, esto es, la distancia desde el origen (0,0) al punto (x, y). Para calcular esta distancia, puedes utilizar el teorema de Pitágoras: la longitud de la hipotenusa de un triángulo será rectángulo es la raíz cuadrada de la suma del cuadrado de los dos catetos del triángulo.

```
console.log(new Vector(1, 2).sumar(new Vector(2, 3)));
// → Vector{x: 3, y: 5}
console.log(new Vector(1, 2).restar(new Vector(2, 3)));
// → Vector{x: -1, y: -1}
console.log(new Vector(3, 4).longitud);
// → 5
```

EJERCICIO: u3e05_vehiculo

Crea una clase denominada **Vehículo** que tenga algunas características como el **color**, **marca**, **modelo** y la **velocidad máxima** que puede alcanzar.

También deberá tener al menos tres métodos además del constructor: **comenzar** que mostrará el mensaje “Encender motor”, **parar** que mostrará “Apagar motor” y **distanciaMax** que recibirá un parámetro con indicando el tiempo de funcionamiento y calculará la distancia máxima que puede haber recorrido, que será el resultado de multiplicar la velocidad máxima por el tiempo.

Crea dos clases más, ambas deben heredar de Vehículo:

- **Coche**: cuya velocidad máxima será 120 y sobrescribirá el método parar para que muestre “Aparcar”.
- **Avión**: cuya velocidad máxima será 1000 y sobrescribirá los métodos comenzar, para que muestre “Despegar”, y parar, para que muestre “Aterrizar”.

Crea las sentencias necesarias para crear objetos y utilizar todos los métodos definidos.

Arrays

EJERCICIO: u3e6_arrayNumeros

Crea un array, cuyo contenido son números.

- Haz una función que muestre en la página los números pares que hay en el array, utilizando un for, utilizando forEach(), for..in y for..of.
- Haz una función que muestre en la página el resultado de sumar todos los elementos del array.
- Haz una función que elimine el último elemento del array, y añada otro elegido por el usuario.
- Haz una función que muestre el valor máximo que hay en el array.

EJERCICIO: u3e7_sumarRango

Crea un programa con las siguientes funciones, y muestra su funcionamiento con ejemplos:

- Crea una función rango, que toma dos argumentos, inicio y fin, y devuelve un array que contiene todos los números desde inicio hasta fin (ambos incluidos).
- Escribe una función suma, que recibe como parámetro un array de números y devuelve la suma de estos números.
- Crea una nueva función rango2, a partir de la función rango, con un tercer argumento opcional que indica el "paso" (step) utilizado para construir el array. Si no se da este argumento, los elementos se incrementan de uno en uno.
  Por ejemplo, rango2(1, 10, 2) devolverá [1, 3, 5, 7, 9].
  También deberá funcionar con números negativos. Por ejemplo, rango2(5, 2, -1) devolverá [5, 4, 3, 2].

EJERCICIO: u3e8_arrayDiscos

Haz un script que gestione una lista de discos, utilizando las funciones para manejar el objeto de tipo disco que has creado en el ejercicio anterior.

Crea un array vacío para almacenar los discos.

Cuando el usuario cargue la página, se mostrarán las opciones:

- Mostrar número de discos.
- Mostrar listado de discos (y le preguntará si quiere mostrarlos en el orden que se encuentran en el array, del revés u ordenados alfabéticamente).
- Mostrar un intervalo de discos (y le pedirá que introduzca el intervalo en formato inicio-fin; luego deberás extraer el valor inicio y fin).
- Añadir un disco (y le preguntará si quiere añadir al principio o al final).
- Borrar un disco (y le preguntará si quiere borrar al principio o al final).
- Consultar un disco (y le preguntará si quiere consultar por posición o por nombre).
- Todas las operaciones que se realicen se irán mostrando en la página con su título.

EJERCICIO: u3e9_dados

Escribir un script que simule el lanzamiento de dos dados. Hacer uso de la función Math.random() para obtener números aleatorios entre 1 y 6 para cada uno de los lanzamientos de los dados. Sumar el resultado de lanzar dos dados y anotar en un array el número de apariciones de dicha suma, repitiendo 36.000 veces esta operación.

EJERCICIO: u3e10_dni

Crea un script en el que se introduzca información sobre los alumnos (nombre, edad, sexo, dni, teléfono,curso,….) y las notas de los mismos en tres asignaturas. Esta información ha de almacenarse en un array de objetos.

Se utilizará la función para comprobar DNI, ya que no se puede dar de alta o modificar un alumno si su DNI es incorrecto.

Para gestionar la información debe permitir realizar las siguientes operaciones:

- Introducir un alumno.
- Consultar datos.
- Borra a un alumno.
- Modifica los datos de un alumno.
- Salir.

EJERCICIO: u3e11_elecciones

Se quieren almacenar los resultados obtenidos en las elecciones a alcalde de Castro Urdiales, y ha habido 4 colegios electorales donde se ha podido votar (ej. Ataulfo Argenta, Zapatero,Santa Catalina, y Riomar) y 6 partidos candidatos a las elecciones (ej. PSOE, PP, Ciudadanos, Podemos y VOX).

Deberás realizar una página web que tendrá varias opciones (puedes implementarla con botones e inputs o prompts).

- Solicitar al usuario todos los votos por colegio electoral y partido, y almacenarlos (el programa indicará el nombre del colegio y del partido, no solo la posición).
- Permitir que el usuario indique un colegio y un partido y pueda modificar el dato almacenado.
- Permitir que el usuario indique un colegio y le muestre el total de votos recibidos.
- Permitir que el usuario indique un partido y le muestre el total de votos recibidos.

Además, en la página web se mostrarán:

- Una tabla con todos los colegios electorales y todos los partidos, y sus votos asociados.
- Calcular el número total de votos por partido y por colegio, y colocarlos al final de las filas y columnas de la tabla.
- Indicar el listado de colegios, ordenados de mayor a menor número de votos.
- Indicar el listado de candidatos, ordenados de mayor a menor número de votos.