Fundamentos de Programación

JavaScript

Arrays, Objetos

IES La Nía

Arrays

- Tipo de datos compuesto que permite agrupar datos.
  - Valores separados por comas, y englobados dentro de corchetes

```
var arrayVacio = [];
var nombreArray = [valor0, valor1, valor2, … valorN-1];
var notas = ['Suspenso', 'Aprobado', 'Bien', 'Notable', 'Sobresaliente'];
```

- Los datos *deberían* ser del mismo tipo, aunque *JavaScript* permite que sean de tipos diferentes.

Representación

```
var notas = ['Suspenso', 'Aprobado', 'Bien',
             'Notable', 'Sobresaliente'];
```

0	Suspenso
1	Aprobado
2	Bien
3	Notable
4	Sobresaliente

Accediendo al array

```
var notas = ['Suspenso', 'Aprobado',
'Bien', 'Notable', 'Sobresaliente'];
```

- Para acceder a un elemento del array, se utiliza la notación 0-index

```
var susp = notas[0];
var apto = notas[1];
```

- También podemos asignar valores en la posición que deseemos (*0-index*):

```
notas[5] = 'Matrícula';
```

- Si accedemos a un índice que no tiene valor, obtendremos `undefined`. En otros lenguajes de programación obtendríamos un error.

Añadiendo contenido

- Asignando un valor a un índice

```
notas[5] = 'Matricula';
```

- Método `push()`

```
notas.push('Matrícula de Honor');
```

  - Inserta el elemento después del último valor

- Método unshift `()`

```
notas.unshift('No Presentado');
```

  - Inserta el elemento como primer valor

Tamaño del Array

- Para obtener su tamaño, podemos acceder a la propiedad **`length`** (igual que con `String`)

```
var tam = notas.length;
notas[tam] = 'Honor';
var nuevoTam = notas.length;
```

Recorrer un array

- Podemos recorrerlo igual que una cadena:

```
for (var i=0; i<array.length; i++) {
  // trabajamos con array[i]
}
```

- O podemos usar un **`for in`**

```
for (var i in array) {
  // trabajamos con array[i]
}
```

Ejercicios

- (`ej401cien`) Rellenar un array de 100 elementos con los números del 1 al 100.
- (`ej402aleatorio`) Rellenar un array con 100 números naturales aleatorios comprendidos entre 0 y 100.
- (`ej403faleatorio`) Crear una función que a partir de una cantidad y un valor máximo, devuelva un array de cantidad elementos rellenado con números aleatorios entre 0 y el valor máximo.

Ejercicios

- (`ej404bola8`) Crea un programa que a cada pregunta responda de manera aleatoria entre un conjunto de respuestas predefinidas, almacenadas en un array:
  - Si, no, quizás, claro que sí, por supuesto que no, no lo tengo claro, seguro, yo diría que sí, ni de coña, etc...

- https://es.wikipedia.org/wiki/Magic_8-Ball

Ejercicios

- (`ej405mates`) Usando la función de `ej403aleatorio`, rellena un array con 100 números aleatorios (entre 0 y 1000), y mostrar:
  - El mayor
  - El menor
  - La media
- (`ej406monedaSH`) Usando un array con los billetes y monedas disponibles, crear una función que a partir de una cantidad, devuelva la cantidad de dinero necesaria de cada tipo.

Problema Olimpiada

Problema 2: Número mínimo de monedas (20 puntos)

Debes implementar un programa que calcule la cantidad mínima de monedas dada una cantidad de céntimos.

El programa deberá preguntar en primer lugar la cantidad de céntimos que se desea calcular. Posteriormente mostrará la cantidad mínima de monedas, indicando las monedas de cada tipo.

Se deben tener en cuenta los siguientes tipos de monedas: 1 céntimo, 2 céntimos, 5 céntimos, 10 céntimos, 20 céntimos, 50 céntimos, 1 euro y 2 euros.

Ejemplo de ejecución:

```
Indique cantidad de céntimos: 487
Monedas necesarias: 7
2 de 2 euros
1 de 50 céntimos
1 de 20 céntimos
1 de 10 céntimos
1 de 5 céntimos
1 de 2 céntimos
```

Ejercicios

- (`ej407dni`) Crear una función que reciba un DNI y devuelta su letra

  El cálculo de la letra del Documento Nacional de Identidad (DNI) es un proceso matemático sencillo que se basa en obtener el resto de la división entera del número de DNI y el número 23. A partir del resto de la división, se obtiene la letra seleccionándola dentro de un array de letras

```
var letras = ['T', 'R', 'W', 'A', 'G', 'M', 'Y', 'F', 'P', 'D', 'X',
'B', 'N', 'J', 'Z', 'S', 'Q', 'V', 'H', 'L', 'C', 'K', 'E'];
```

Ejercicios

- (`ej408vectorPorEscalar`) Crea una función que reciba un vector y un escalar y devuelve su producto (un nuevo vector).
- (`ej409sumaVector`) Crea una función que reciba dos vectores numéricos y devuelva un vector con la suma de ambos
- (`ej410productoVector`) Crea una función que reciba dos vectores numéricos y devuelva el producto escalar (un número).

Ejercicios

- (`ej411arraySexos`) Crea una función que a partir de un array rellenado con ('M','F'), devuelva mediante un array cuantos hombres y cuantas mujeres hay (por ejemplo, [23,43]).
- (`ej412arrayPares`) Crea una función que reciba un array rellenado de númerosh y devuelva un nuevo array con solo los números pares del array recibido.
- (`ej413quiniela`) Crea una función que devuelva una combinación de la quiniela

Eliminando un elemento

- **`delete array`**`[`*`indice`*`]` → eliminar el elemento de la posición *índice*.
  - El resto de elementos se quedan donde estaban

```
var apto = notas[1];
delete notas[1];
var vacio = notas[1];  // undefined
```

Métodos de borrado

- **`pop()`** → elimina el último elemento
- **`shift()`** → elimina el 1er elemento. El resto de elementos se mueven

```
var notas = ['Suspenso', 'Aprobado', 'Bien', 'Notable',
'Sobresaliente'];

var detras = notas.pop();
console.log(detras); // "Sobresaliente"
console.log(notas); // ["Suspenso", "Aprobado", "Bien", "Notable"]

var delante = notas.shift();
console.log(delante); // "Suspenso"
console.log(notas);  // ["Aprobado", "Bien", "Notable"]
```

Ejercicios

- (`ej414listaEmpleados`) Crea un aplicación que permita listar, insertar y borrar empleados, haciendo uso de arrays.

  Se mostrará un menú al usuario.

  Al listar los empleados, para cada empleado se mostrará su número (posición en el array), y su nombre.

  Los usuarios se borrarán por nombre o por número.

Ejercicios

- (`ej415ganador`) Crea una aplicación que pida nombre de personas hasta introducir un nombre en blanco.

  En ese momento, el programa debe elegir una persona de manera aleatoria, la cual será la ganadora, y se mostrará su nombre por pantalla.

Cadenas como *Arrays*

- Podemos acceder a los elementos de una cadena mediante la notación de array.

```
var heroe = 'Batman';
var b = heroe[0];
var a = heroe[1];
```

- `cadena.`**`split`**`(separador)` → divide una cadena en fragmentos a partir de un separador y la devuelve como un array

Ejercicios

- (`ej416vocales`) Crea una función que a partir de una cadena devuelva el número de vocales que tenga.
- (`ej417ordenNombre`) Crea una función que a partir de una cadena con "Apellidos;Nombre", devuelva "Nombre Apellidos"
- (`ej418fechaEuropa`) Crea una función que a partir de una cadena con una fecha en formato americano (M/D/Y), la devuelva en formato europeo (D/M/Y)

Ejercicios

- (`ej419filtroValores`) Crea un programa que permita al usuario leer un conjunto de números separados por espacios. El programa filtrará los números leídos para volver a mostrar únicamente los números pares (ver `ej412arrayPares`) e indicará la cantidad existente.

```
Dame números: 1 4 7 9 23 10 8
Los 3 números pares son: 4 10 8
```

Métodos mutadores I

- **`reverse()`** → invierte el orden de los elementos
- **`sort()`** → ordena alfabéticamente los elementos
  - Númericos → `array.sort(function(a, b){return a-b;});`

```
var notas = ['Suspenso', 'Aprobado', 'Bien', 'Notable', 'Sobresaliente'];

notas.reverse();
console.log(notas);  // ["Sobresaliente", "Notable", "Bien", "Aprobado", "Suspenso"]

notas.sort();
console.log(notas);  // ["Aprobado", "Bien", "Notable", "Sobresaliente", "Suspenso"]
```

Ejercicios

- (`ej420medianaArray`) Crea una función que a partir de un array con números devuelva su mediana (*elemento que ocupa la posición central dentro de un array ordenado*)
- (`ej421mediaTruncadaArray`) Crea una función que obtenga la media truncada eliminando el valor más bajo y más alto de un array
- (`ej422desordenarArray`) Crea una función que desordene un array

Métodos mutadores II

- **`splice(inicio, cantidad [,elem1, ..[,elemN]])`** → permite eliminar y añadir contenido simultáneamente

```
var notas = ['Suspenso', 'Aprobado', 'Bien', 'Notable',
'Sobresaliente'];

notas.splice(1, 2, "Apto");
console.log(notas); // ["Suspenso", "Apto", "Notable",
"Sobresaliente"]
```

Ejercicios

- (`ej423frecuencia`) Crear una función que a partir de un array rellenado con número devuelva un array con la frecuencia de los números.
- (`ej424moda`) Crear una función que a partir de un array con números del 1 al 100, devuelva su moda

Arrays Bidimensionales

- Array de Arrays
- Estructura de datos tabular

```
var menu = [["Macarrones", "Merluza", "Fruta"],
            ["Arroz", "Pechuga", "Yogur"],
            ["Lentejas", "Calamares", "Fruta"],
            ["Caldo", "Pelota", "Yogur"],
            ["Ensalada", "Guisado", "Helado"]];

var postresViernes = menu[4][2];
```

- Se recorren con dos bucles anidados

Ejercicios

- (`ej425mostrarMatriz`) Crea una función que muestre por consola una matriz, mostrando cada fila en una linea separada.
- (`ej426mostrarVertices`) Crea una función que muestre por consola los elementos que son vértices.

Creando una matriz

```
function rellenarMatriz(longitud) {

  var matriz = [[]];

  for (var i=0;i<longitud;i++) {
    matriz[i] = [];
    for (var j=0;j<longitud;j++) {
      matriz[i][j] = 0;
    }
  }

  return matriz;
}
```

Ejercicios

- (`ej427MatrizHBX`) Crear una función que rellene una matriz de altura H y base B, con un carácter indicado por el usuario.
- (`ej428MatrizIdentidad`) Crear una función que rellene una matriz de tamaño N con la identidad (diagonal principal a 1, resto a 0)

Ejercicios

- (`ej429MatrizDiagonalSecundaria`) Crear una función que rellene una matriz de tamaño N con la diagonal secundaria a 1, el resto a 0.
- (`ej430MatrizMas`) Crear una función que rellene una matriz de tamaño N con la fila y la columna central a 1, el resto a 0.
- (`ej431MatrizTranspuesta`) Crear una función que a partir de una matriz, devuelva su matriz transpuesta.

Ejercicios

- (`ej432MatrizTriangulo`) Crear una función que rellene una matriz de tamaño N con la diagonal y los elementos inferiores a 1, el resto a 0
- (`ej433MatrizCuadrado`) Crear una función que rellene una matriz de tamaño N con los bordes a 1 y el resto a 0.

Ejercicios

- (`ej434MatrizNumerica`) Crear una función que rellene una matriz de tamaño N con números consecutivos. Al llegar al 9, volverá a comenzar por el 0.

```
1 2 3 4
5 6 7 8
9 0 1 2
3 4 5 6
```

- (`ej435MatrizAjedrez`) Crear una función que rellene una matriz de tamaño N de manera similar a un tablero de ajedrez.

```
1 0 1 0 1
0 1 0 1 0
1 0 1 0 1
0 1 0 1 0
1 0 1 0 1
```

Proyecto – Hundir la Flota

- Tablero 8x8
- Rellenar con 0s (agua) → `crearTablero()`
- Función que recibe el tablero, coordenadas y devuelve 0 (agua), 1 (tocado), 2 (hundido) → jugada(tablero, fila, col)
- Función que recibe el tablero y el barco (2, 3, 4) y lo coloca de manera aleatoria.

Objeto

- Contenedor de propiedades
- Cada propiedad tiene un nombre y un valor
- Un objeto puede contener otros objetos
- Operador .
- Se crean dinámicamente

```
var persona = new Object();
persona.nombre = "Aitor";
persona.apellido1 = "Medrano";
```

Método

- Operaciones de un objeto
- Función anónima asignada a una propiedad
- Dentro de los métodos, this referencia al objeto

```
persona.getNombreCompleto = function() {
  return this.nombre + " " + this.apellido1;
}
console.log(persona.getNombreCompleto());
```

Objeto Literal

- Notación más sencilla → similar a JSON
- Par de llaves que rodean 0 o más parejas de clave:valor separadas por comas
  - cada clave puede ser una propiedad o un método

```
var nadie = {};
var persona = {
  nombre : "Aitor",
  apellido1 : "Medrano",
  getNombreCompleto : function() {
    return this.nombre + " " + this.apellido1;
  }
};
```

Objetos Anidados

- Cualquier propiedad puede a su vez ser un objeto
  - Agrupa información

```
var cliente = {
  nombre: "Bruce Wayne",
  email: "bruce@wayne.com",
  direccion: {
    calle: "Mountain Drive",
    num: 1007,
    ciudad: "Gotham"
  }
};
```

```
var cliente = {};
cliente.nombre = "Bruce Wayne";
cliente.email = "bruce@wayne.com";
cliente.direccion = {};
cliente.direccion.calle = "Mountain Drive";
cliente.direccion.num = 1007;
cliente.direccion.ciudad = "Gotham";
```

Ejercicios

- (`ej436ObjetoNombre`) Crear un objeto literal con tus datos personales.
- (`ej437Fecha`) Crear un objeto fecha compuesto de día, mes y año.
- (`ej438FechaMetodo`) A partir del objeto Fecha, añade un método para calcular la diferencia de días con la fecha actual.

https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/Date

Ejercicios

- (`ej439FechaFinSemana`) A partir del objeto Fecha, añadir un método que averigüe si es fin de semana.
- (`ej440FechaEstacion`) A partir del objeto Fecha, añadir un método que devuelva la estación del año.

Función Constructor

- Función que construye un objeto
  - `this` referencia al propio objeto

```
var Persona = function(nombre, apellido1) {
  this.nombre = nombre;
  this.apellido1 = apellido1;

  this.getNombreCompleto = function() {
    return this.nombre + " " + this.apellido1;
  };
};
```

```
var persona = new Persona("Aitor", "Medrano");
```

Ejercicios

- (`ej441FechaFuncionConstructor`) Reescribir el objeto `Fecha` mediante una función constructor.
- (`ej442Pieza`) Crea un objeto `Pieza` compuesto de
  - Tipo de Barco: 2, 3, 4
  - Estado: 0, 1 (0 agua, 1 tocado)
  - Orientación: 0,1 (0 horiz, 1 vertical)
  - Siguiente: [x,y]
  - Anterior: [x,y]