JavaScript 2023

Práctica 2

Objetivos

- Operadores, estructuras de datos y de control.
- Funciones.
- Introducción a la manipulación del documento.
- Introducción al manejo de eventos.

Material Útil de Referencia

- https://developer.mozilla.org/es/docs/Web/JavaScript

Ejercicio 1

Dadas las siguientes definiciones de variables, imprima en la consola el resultado que arroja el operador `typeof` para cada una.

```
let a = 1;
let b = true;
let c = "Hola";
let d = null;
let e;
let f = 2n ** 60n;
let g = new Date();
let h = [1,2,3,4];
let i = 'Hola';
let j = { x: 2.0, y: -3.6 };
let k = /^[a-zA-Z0-9.!#$%&'*+/=?^_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-
9-]+)*$/;
```

Por ejemplo:

```
console.log(typeof a);
```

Ejercicio 2

En cada invocación console.log, analice qué valor que debería imprimir. Luego verifique ejecutando el script.

```
let a = 25;
console.log(++a);
console.log(a++);
```

```
console.log(a == '27');
console.log(a === '27');
```

Ejercicio 3

Defina las funciones max, min, avg, sum de manera que reciban un arreglo de números como parámetro.

```
function max(values) {
    // retorna el valor máximo
}
function min(values) {
    // retorna el valor mínimo
}
function avg(values) {
    // retorna el valor promedio
}
function sum(values) {
    // retorna la sumatoria de los valores
}
```

¿Cómo puede manejar el caso en que el arreglo no tenga ningún elemento?

¿Cómo puede manejar el caso en que el arreglo contenga un elemento que no sea numérico?

Realice pruebas con diferentes valores.

```
var prices = [1.2, 2, 22, -33, 12, 0.0, "13", Math.PI];
var names = ["John", "Paul", "George", "Ringo"];

console.log(max(prices));
console.log(min(prices));
console.log(avg(prices));
console.log(sum(prices));

console.log(max(names));
console.log(min(names));
console.log(avg(names));
console.log(sum(names));

console.log(max([]));
console.log(min([]));
console.log(avg([]));
console.log(sum([]));
```

Ejercicio 4

Dada la siguiente función `concat` que devuelve la concatenación de dos strings,

```
function concat(left, right) {
    return left.concat(right);
}
```

¿qué imprime la siguiente sentencia?

```
var names = ["John", "Paul", "George", "Ringo"];

console.log(names.reduce(concat));
```

Resuelva el ejercicio 3 utilizando reduce, Math.max y Math.min.

¿Qué sucede si se invoca `reduce` sobre un arreglo vacío?

Ejercicio 5

Implemente la función de igualdad entre arreglos y realice las pruebas que se indican.

```
function arrayEquals(a, b) {
    // implementar
}

var a = [1,2,3,4];
var b = [1,2,3,4];
var c = [1,2,3,4,5];
var d = "Hola";
var e = null;

console.log(arrayEquals(a, a));
console.log(arrayEquals(a, b));
console.log(arrayEquals(b, c));
console.log(arrayEquals(e, c));
console.log(arrayEquals(c, d));
console.log(arrayEquals(e, e));
```

¿Utilizó `==` o `===` para comparar los elementos?

Nota: dos arreglos son iguales si tienen la misma longitud y elementos iguales en cada posición.

Ejercicio 6

Implemente la función `isInteger` que reciba un parámetro y que retorne si es un número entero.

```
isInteger(2); // retorna true
isInteger(2.0); // retorna true
isInteger(2.1); // retorna false
```

```
isInteger(-10); // retorna true
isInteger([1]); // retorna false
isInteger("2"); // retorna false
isInteger(true); // retorna false
isInteger(null); // retorna false
var num;
isInteger(num); // retorna false
```

Ejercicio 7

Implemente una función que reciba un String y retorne si es numérico.

```
isNumeric("2") // retorna true
isNumeric("2a") // retorna false
isNumeric(2) // retorna false
```

Pista: Puede usar la función isNaN

Ejercicio 8

Dadas las siguientes asignaciones.

```
var prices = {
    MILK: 48.90,
    BREAD: 90.50,
    BUTTER: 130.12
};

var amounts = {
    MILK: 1,
    BREAD: 0.5,
    BUTTER: 0.2
}
```

Verifique qué devuelven las siguientes expresiones

```
console.log(typeof prices);
console.log(prices.BREAD);
console.log(amounts["MILK"]);
```

Implemente la función `total` que retorna el valor total de `amounts` de acuerdo a los valores de `prices` suponiendo que `prices` tiene los precios por unidad (kilo o litro) y `amounts` tiene la cantidad comprada de cada producto.

```
console.log(total(prices, amounts)); // imprime 120.174
```

Verifique el resultado de la siguiente invocación.

```
var amounts2 = {
    BREAD: 1.5
};
console.log(total(prices, amounts2));
```

Realice los cambios que considere pertinentes si el resultado no es el esperado.

Pista: Object.keys(), Object.entries()

Ejercicio 9

Dado el siguiente arreglo de palabras, imprímalo en orden alfbético y en orden alfabético inverso.

```
var words = ['perro', 'gato', 'casa',
    'árbol', 'nube', 'día', 'noche',
    'zanahoria', 'babuino'];
```

Verifique que con su implementación la palabra `árbol` quede ubicada en la posición correcta.

```
var atoz = [ "árbol", "babuino", "casa", "día", "gato",
  "noche", "nube", "perro", "zanahoria"]

var ztoa = ["zanahoria", "perro", "nube", "noche", "gato",
  "día", "casa", "babuino", "árbol"];
```

Pista: puede usar la función localeCompare.

Ejercicio 10

Dado el arreglo de palabras del ejercicio 9, imprímalo ordenado por la longitud de cada palabra de manera que la primera palabra sería `día` y la última `zanahoria`.

Ejercicio 11

Dadas las sentencias que siguen a continuación,

```
function identity(x){
    return x;
}
var id = function(x){
    return x;
}
```

```
var iden = x => x;

var identidad = identity;
```

compruebe el typeof de cada una.

```
console.log(typeof identity);
console.log(typeof id);
console.log(typeof iden);
console.log(typeof identidad);
```

Compruebe el resultado de las siguientes sentencias.

```
console.log(identity('Hola'));
console.log(id('Hello'));
console.log(iden('Buen día'));
console.log(identidad('Buen día'));
```

Ejercicio 12

Implemente una función equivalente al método reduce de los Arrays (ver ejercicio 4).

```
function reduce(array, binaryOperation, initialValue){
    // TODO: implementar
}
```

Por ejemplo las siguientes invocaciones a `numbers.reduce` y `reduce` deben retornar el mismo valor.

```
var numbers = [ 1, 3, 4, 6, 23, 56, 56, 67, 3,
    567, 98, 45, 480, 324, 546, 56 ];
var sum = (x, y) => x + y;
numbers.reduce(sum, 0);
reduce(numbers, sum, 0);
```

Ejercicio 13

Dadas las siguientes definiciones de objetos

```
var alice = {
    name : "Alice",
    dob : new Date(2001, 3, 4),
    height : 165,
    weight : 68
```

```
};
var bob = {
    name : "Robert",
    dob : new Date(1997, 0, 31),
    height : 170,
    weight : 88
};
var charly = {
    name : "Charles",
    dob : new Date(1978, 9, 15),
    height : 188,
    weight : 102
};
var lucy = {
    name : "Lucía",
    dob : new Date(1955, 7, 7),
    height : 155,
    weight : 61
};
var peter = {
    name : "Peter",
    dob : new Date(1988, 2, 9),
    height : 165,
    weight : 99
};
var luke = {
    name : "Lucas",
    dob : new Date(1910, 11, 4),
    height : 172,
    weight : 75
};
```

1. Implemente una función que devuelva un arreglo con los nombres de las personas con un IMC mayor a 25.
2. Implemente una función que devuelva un arreglo de las edades de las personas indexado por el nombre de cada una. (Por ejemplo algo de la forma `["Bobby": 22, "Mark": 36]`).
3. Implemente una función que devuelva un arreglo con el IMC de los mayores de 40.
4. Implemente una función que devuelva el IMC promedio de todas las personas.
5. Implemente una función que devuelva a la persona más joven.
6. Implemente una función que devuelva un arreglo de personas ordenado por estatura.

¿Pudo usar find, reduce, filter o map para resolver alguna de las operaciones?

Ejercicio 14

Dada la función `showMessage` que sigue,

```
function showMessage(message){
    alert(message);
}
```

cree un documento html que tenga un párrafo como el que sigue

```
    <p onclick="showMessage('Hola')">Por favor haga click aquí</p>.
```

y compruebe que se muestre el mensaje al hacer click en el párrafo.

Ejercicio 15

Cree un documento html con un formulario.

```
    <html>
    <head>
        ...
    </head>
    <body>
        <form>
            <input type="text" id="texto"/>
            <input type="button" value="Enviar" onclick="enviar()"/>
        </form>
    </body>
    </html>
```

Defina la función `enviar` para que obtenga el valor del texto ingresado en el `input` y lo imprima en la consola.

```
function enviar(){
    // TODO implementar usando document.getElementById('texto').value;
}
```

Agregue otro `input` similar al existente y modifique la función `enviar` para que al hacer click en el botón el texto del primer `input` se copie al segundo `input`.

Ejercicio 16

Cree un documento html con 1 input, 3 imágenes ocultas con id imagen-1, imagen-2 e imagen-3 y un botón "Mostrar".

Cuando el usuario escribe el nombre de la imagen en el input (imagen 1, imagen 2 o imagen 3) y hace click en el botón, se debe mostrar la imagen que corresponde.

¿Qué pasa si no hay ningún texto ingresado el input? ¿Cómo lo resuelve?

¿Qué pasa si el texto ingresado no coincide con ninguna imagen? ¿Cómo lo resuelve?

Ejercicio 17

Modifique el ejercicio anterior usando otro elemento HTML para evitar que el usuario tenga que escribir el nombre de las imágenes.

Ejercicio 18

En un formulario con un select como el que sigue,

```
<select>
    <option value="red">Rojo</option>
    <option value="green">Verde</option>
    <option value="white">Blanco</option>
    <option value="blue">Azul</option>
</select>
```

implemente una función que al hacer click en un botón modifique el color de fondo del documento de acuerdo al valor seleccionado.

Pista: puede agregar un atributo `id` tanto en en el select como en el body.

¿Cómo puede hacer que al cargar la página el valor seleccionado sea siempre "Blanco"?

Ejercicio 19

Modifique el ejercicio anterior para que no haya ningún botón y el color cambie apenas se modifica la selección.

Use el evento load para asegurarse que al cargar la página el color de fondo coincida con el seleccionado en el select.

Entrega Obligatoria

Junto a los demás integrantes de su grupo, implemente **los ejercicios 16, 17 y 18 (obligatorios) y 19 (opcional)** y suba el código al repositorio git asignado por la cátedra.

Consideraciones sobre la entrega

- Se debe realizar en grupo.
- Se debe subir al repositorio git asignado al grupo en gitlab.
- Será evaluado.
- Cada integrante del grupo debe hacer sus *commits* con su aporte a la solución.