ACADEMY by NUMEN

FULL STACK

Comenzamos en unos minutos


ACADEMY
by NUMEN

Javascript: Clase 1

Revisemos la tarea de la clase pasada!

ACADEMY by NUMEN

Tarea sobre coerción de datos

¿Cuál crees que será el resultado de la ejecución de estas operaciones?

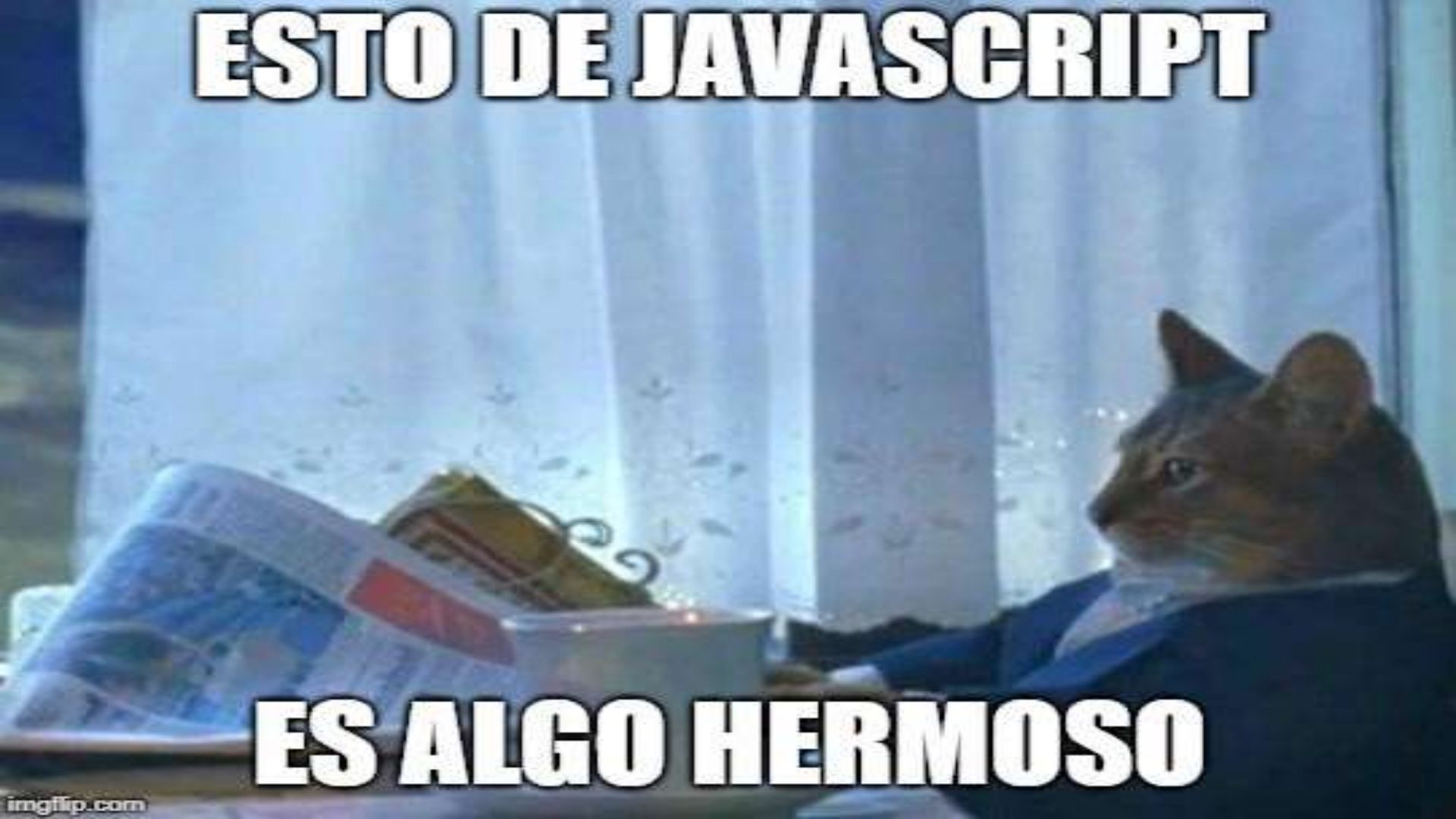
```
6 / "3" =
"2" * "3" =
4 + 5 + "px" =
"$" + 4 + 5 =
"4" - 2 =
"4px" - 2 =
7 / 0 =
parseInt("09") =
```

```
3>2>1 =
12 + ""  =
"15" * 2  =
"15" - "11" =
undefined + 6 =
"Hello" + null =
null + 25 =
```

```
true + true =
false + 10=
5 && 2 =
2 && 5 =
5 || 0 =
0 || 5 =
true && false =
```

```
false || !false =
!2 =
"texto" || 0 =
2 || "prueba“ =
```

Piensa primero cuál será el resultado y después prueba ver la respuesta usando la siguiente pagina: https://jsconsole.com/


ESTO DE JAVASCRIPT
ES ALGO HERMOSO
imgflip.com

Javascript: Clase 2

ACADEMY by NUMEN

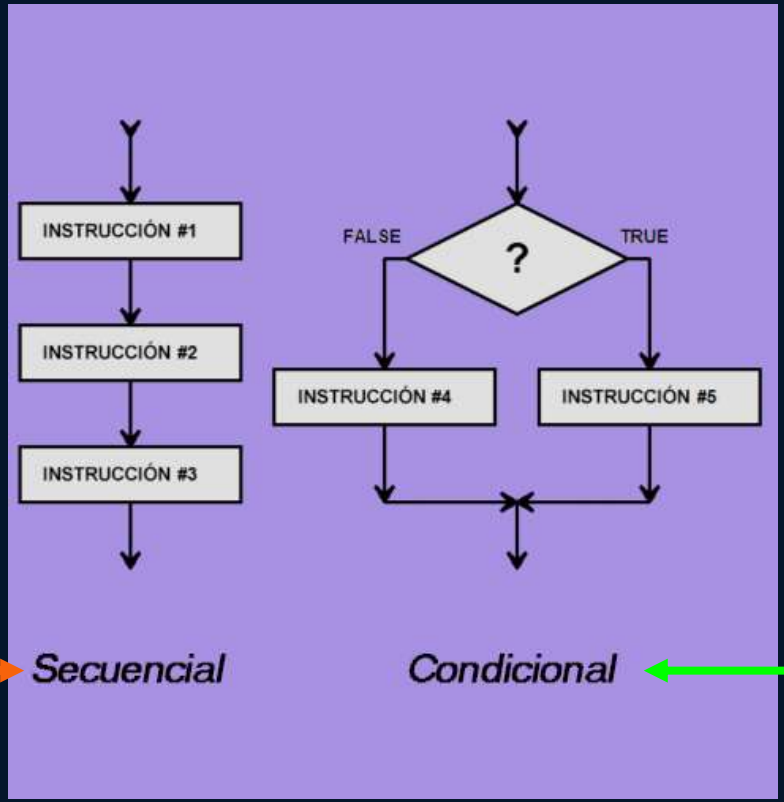
Estructuras de Control de Flujo

ACADEMY by NUMEN

Estructura secuencial:

Es aquella estructura en la cual una instrucción o acción sigue a otra en un único hilo.

Por ejemplo el procedimiento para lavarse los dientes.



Estructura condicional:

Esta estructura nos permite decidir por cuál alternativa seguirá el flujo del programa dependiendo del resultado de la evaluación de una condición.

Estructura secuencial

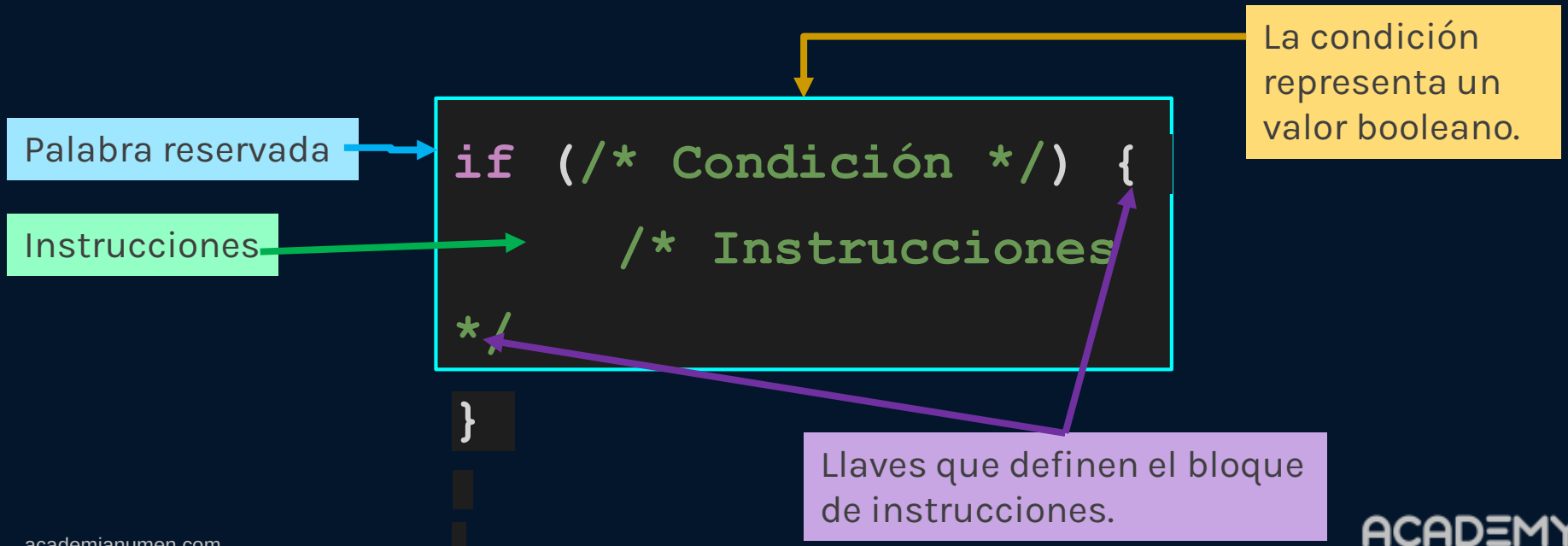
Una hoja de JavaScript funciona como un gran algoritmo, en el cual cuanta instrucción coloquemos en ella se irá ejecutando en un orden secuencial, es decir una detrás de otra. Aquí podemos observar una cadena de impresiones en la consola.

```
console.log("Inicio")
console.log("Instrucción 1")
console.log("Instrucción 2")
console.log("Instrucción 3")
console.log("Instrucción 4")
console.log("Instrucción 5")
console.log("Fin")
```

Estructura condicional o de control

If - else

Para poder representar una estructura condicional en JavaScript necesitamos de una serie de palabras reservadas que definirán algunos bloques cerrados donde depositaremos una o varias instrucciones a ejecutar.



Veamos cómo transformamos una estructura secuencial en una estructura condicional.

```
var comida = 1;

console.log("No tengo hambre");

if (comida === 0) {
    console.log("Tengo hambre, voy a comer");
}

console.log("No tengo hambre");
```

else

else es una palabra reservada del lenguaje que encierra instrucciones que se ejecutan cuando la condición de nuestra estructura no se cumple.

```
var numero = 5;

if (numero > 10) {
    console.log("Mayor que 10")
} else {
    console.log("Menor que 10")
}
```

else if

Para poder extender nuestra estructura a múltiples condiciones utilizamos el else if. Estos pueden encadenarse infinitamente, motivo por el cual, si escala demasiado se vuelve algo complejo de leer.

```
var numero = 1;

if (numero === 1) {
    console.log("Instrucción 1")
} else if(numero === 2) {
    console.log("Instrucción 2")
} else if(numero === 3) {
    console.log("Instrucción 3")
} else if(numero === 4) {
    console.log("Instrucción 4")
} else if(numero === 5) {
    console.log("Instrucción 5")
} else {
    console.log("Instrucción X")
}
```


E480
Backspace
Enter
Shift
if
else if
else if
else if
else

Estructuras de control de flujo: Switch - Case

La declaración switch evalúa una expresión, comparando el valor de esa expresión con una instancia case, y ejecuta declaraciones asociadas a ese case, así como las declaraciones en los case que siguen.

```
var productos = 2;

switch (productos) {
    case 1:
        console.log("Té")
        break;
    case 2:
        console.log("Café")
        break;
    case 3:
        console.log("Agua")
        break;
    default:
        console.log("Caramelo media hora")
        break;
}
```

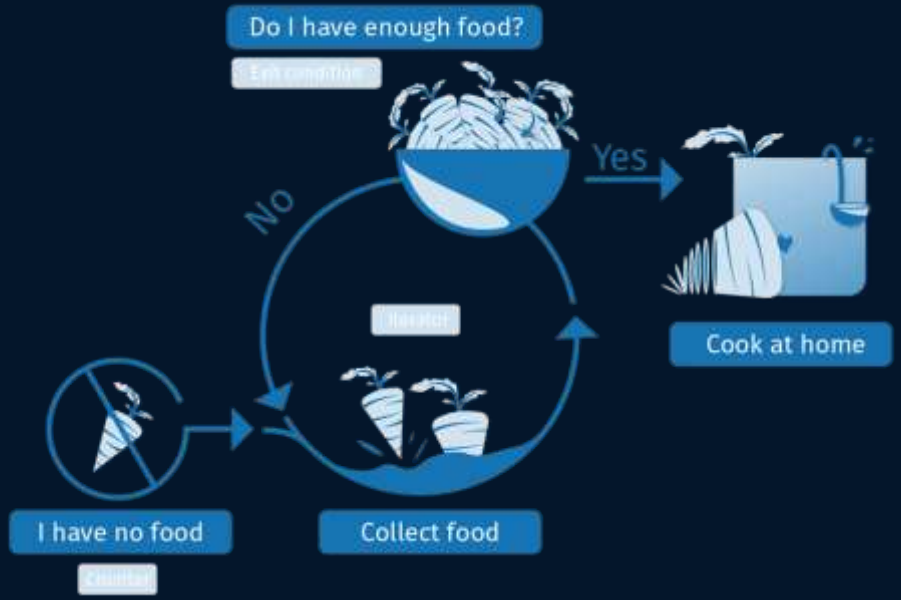

if else
switch statements
confusion about what to choose

Bucles

ACADEMY by NUMEN

¿Qué es un bucle o ciclo?

Los bucles ofrecen una forma rápida y sencilla de hacer algo repetidamente. Hay muchos diferentes tipos de bucles, pero esencialmente, todos hacen lo mismo: repiten una acción varias veces. Los diversos mecanismos de bucle ofrecen diferentes formas de determinar los puntos de inicio y terminación del bucle. Hay varias situaciones que son fácilmente atendidas por un tipo de bucle que por otros.



Podríamos pensar el bucle como una carrera. Mientras que no se cumpla la condición de que estemos cansados podremos seguir corriendo. Pero en el momento que esta se cumple, nos detendremos.

Bucles

while

Crea un bucle que ejecuta una sentencia especificada mientras cierta condición se evalúe como verdadera. Dicha condición es evaluada antes de ejecutar la sentencia.

```
var contador = 0;

while (contador < 10) {
    console.log(contador);
    contador++
}
```

do while

La sentencia (hacer mientras) crea un bucle que ejecuta una sentencia especificada, hasta que la condición de comprobación se evalúa como falsa. La condición se evalúa después de ejecutar la sentencia, dando como resultado que la sentencia especificada se ejecute al menos una vez

```
var contador = 0;

do {
    console.log("do while" + contador);
    contador++
} while (contador < 10);
```


while (not edge) {
run();
}
do {
run();
} while (not edge);

Palabra reservada

Variable que contiene el valor inicial del bucle.

Condición que, en caso de dejar de cumplirse, detendrá el bucle.

Incrementador que irá acercando el valor de la variable a la condición de corte.

```
for (/* Variable */; /* Condicion */; /* Incrementador */) {
    /* Instrucciones */
}
```

Instrucciones a ejecutar de manera repetitiva.

Veamos cómo se vería en un bucle for el contador que hicimos en los bucles anteriores.

```
for (var contador = 0; contador < 10; contador++) {
    console.log(contador)
}
```

Construyendo un acumulador de manzanas

```
var canasta = 0;

for (var manzanas = 1; manzanas <= 10; manzanas++)
{
    canasta = canasta + 1
}

console.log(canasta)
```

Ahora enfrentemos algunos zombies

```
var balas = 10;

for(var zombie = 10; zombie > 0 ; zombie--) {
    balas = balas - 1
}

if (balas > -1) {
    console.log("You survived, mission complete")
} else {
    console.log("You died, mission failed")
}
```

```
for (int i = 0; i < 3; i++)
```

i = 0

i = 1

i = 2

i = 3

Introducción a Funciones

Funciones

Una **función** es un bloque de código, autocontenido, que se puede definir una vez y ejecutar en cualquier momento.
Las funciones en JS son objetos, un tipo especial de objetos:
Se dice que las funciones son ciudadanos de primera clase porque pueden asignarse a un valor, y pueden pasarse como argumentos y usarse como un valor de retorno.

```
// Declaración de función
function miPrimeraFuncion() {
    console.log("1");
    console.log("2");
    console.log("3");
}

// Llamado a la función
miPrimeraFuncion();
```

Estructura básica de una función

Palabra reservada que define la estructura

Nombre de la función

Palabra reservada que indica que valor retornará la función al finalizar el algoritmo.

```
function suma( a, b ) {
    return a + b;
}

suma();
```

Los parámetros son la puerta de entrada de datos a una función.

El llamado de la función es el interruptor de ejecución de la función.

EJEMPLOS DE FUNCIONES

Analicemos con detenimiento las siguientes dos funciones y respondamos las preguntas:

```
// Declaración de función con un parámetro
function devolverString(str) {
          return str;
}

// Llamado a la función
devolverString('Hola');
```

```
// Declaración de función con dos parámetros
function suma(a, b) {
    return a + b;
}

// Llamado a la función
suma(10,5);
```

- La función `devolverString`, ¿recibe algún valor? ¿Cuál?
- La función `devolverString`, ¿regresa algún valor? ¿Cuál?
- ¿Puedo cambiar el valor que `devolverString` recibe?¿ Como?
- ¿Puedo cambiar el valor que `devolverString` regresa? ¿Como?
- Mismas preguntas para la función `suma`

Veamos ahora cómo podemos usar la estructura IF en una de las funciones que armamos anteriormente

```
// Para corroborar si efectivamente A y B eran iguales o diferentes, usamos el operador de comparación "==="

function sonIguales( a, b ) {
 return a === b
}

sonIguales(5,5);
```

```
// Ahora, usaremos la estructura de control IF para hacer exactamente lo mismo.

function sonIguales( a, b ) {
    if (a === b) {
          return true
    } else {
          return false
    }
}

sonIguales(5,5);
```

¡Ahora es tu turno de programar funciones! ¡Leé con detenimiento la consigna y resuelve ambas funciones!

```
function resta( x , y ) {
 // Resta "y" de "x", y devuelve el valor.
}

resta();
```

```
function sonIguales( a, b ) {
// Devuelve TRUE si "a" y "b" son guales. De lo contrario, devuelve FALSE
}

sonIguales();
```

ACADEMY
by NUMEN

Ahora, hablemos sobre la tarea para la próxima clase!

ACADEMY by NUMEN