

# PROGRAMACIÓN IMPERATIVA EN JAVASCRIPT





¡HOLA!

# Soy Agustín Curto

Licenciado en Ciencias de la Computación  
Facultad de Matemática, Astronomía, Física y  
Computación  
Universidad Nacional de Córdoba

[agucurto95@unc.edu.ar](mailto:agucurto95@unc.edu.ar)

# SOBRE EL CURSO: FORMALIDADES

- 2 meses de cursado, del 16/5 al 6/7
- Teóricos: martes y jueves de 18:30 a 20:30 hs
- Prácticos: martes y jueves de 20:30 a 21:30 hs. Sábados por la mañana, horario a coordinar con su ayudante.
- 14 clases (25/5 y 20/6 feriados)
- N prácticos, se aprueba con el 70%, N\*.7
- Aula virtual (grupo de google por ahora)
- Basado en el curso de Mozilla, con ayudita de chatGPT :D

# SOBRE EL CURSO: OBJETIVOS

- Es una breve introducción a la programación y a JavaScript
- Les puede servir
  - Para determinar si programar es algo que les gusta
  - Como herramienta para su trabajo diario
  - Como un hobby
  - De disparador para estudiar una carrera de programación
    - Carreras de programación en Argentina
- No es la idea del curso conseguir un trabajo afín al finalizar

# PROGRAMA

1. ENTORNO DE PROGRAMACIÓN
2. PROGRAMAR EN EL EDITOR
3. VARIABLES, CONSTANTES Y SUS TIPOS
4. OPERACIONES BÁSICAS
5. CONDICIONALES
6. CICLOS
7. FUNCIONES

# 1. ENTORNO DE PROGRAMACIÓN



# ¡QUÉ ES UN PROGRAMA?

- Conjunto ordenado de instrucciones que le dice a la computadora qué hacer
- Se escriben en un lenguaje de programación
- Los utilizamos en nuestra vida cotidiana
- Ejemplos:
  - WhatsApp
  - Google (buscador/página)
  - Word
  - Netflix
  - etc.

# ¿QUÉ ES PROGRAMAR?

- Proceso de escribir instrucciones que conforman un programa
- Esas instrucciones se escriben en un lenguaje de programación (legible por el humano)
- El programa se traduce a código binario (el lenguaje que hablan las computadoras)

# ¿QUÉ ES JAVASCRIPT?

- Es un lenguaje de programación
- Es interpretado
- Es el estándar web (front-end)
- Se ejecuta en el navegador (chrome, firefox, edge)
- Ha tomado gran poder del lado de los servidores (back-end)
- Será el foco de este curso

# LENGUAJE COMPILADO VS. INTERPRETADO

## Compilado

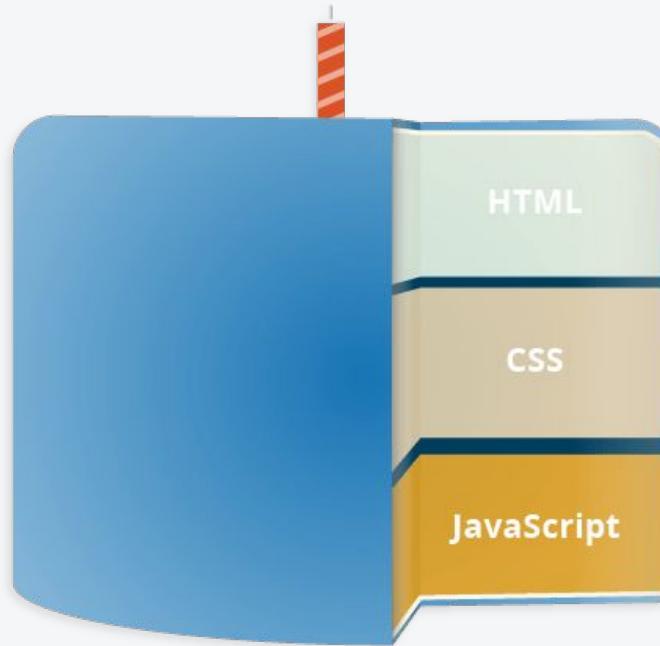
- Se traduce directamente a código de máquina
- El proceso de traducción se realiza antes de la ejecución, se llama *compilación*
- Ejemplos: C++, Java, Swift

# LENGUAJE COMPILADO VS. INTERPRETADO

## Interpretado

- Se ejecuta línea por línea
- No se compila antes de la ejecución
- El código fuente se traduce a código máquina en tiempo real
- Ejemplos: JavaScript, python, ruby

# LA TORTA DE LAS TECNOLOGÍAS WEB



# EJEMPLO: HTML

TEST.HTML

```
<p>Player 1: Chris</p>
```

# EJEMPLO: HTML + JAVASCRIPT

TEST.HTML

```
<p>Player 1: Chris</p>
```

TEST.JS

```
const para = document.querySelector('p');

para.addEventListener('click',
updateName);

function updateName() {
  let name = prompt('Enter a new name');
  para.textContent = 'Player 1: ' + name;
}
```

# EJEMPLO: HTML + CSS + JAVASCRIPT

TEST.HTML

```
<p>Player 1: Chris</p>
```

TEST.CSS

```
p {  
    font-family: 'helvetica neue',  
    helvetica, sans-serif;  
    letter-spacing: 1px;  
    text-transform: uppercase;  
    text-align: center;  
    border: 2px solid  
    rgba(0,0,200,0.6);  
    background: rgba(0,0,200,0.3);  
    color: rgba(0,0,200,0.6);  
    box-shadow: 1px 1px 2px  
    rgba(0,0,200,0.4);  
    border-radius: 10px;  
    padding: 3px 10px;  
    display: inline-block;  
    cursor: pointer;  
}
```

TEST.JS

```
const para = document.querySelector('p');  
  
para.addEventListener('click',  
updateName);  
  
function updateName() {  
    let name = prompt('Enter a new name');  
    para.textContent = 'Player 1: ' + name;  
}
```

LIVE CODE

# ALGUNOS EJEMPLOS

- Antes que nada hablemos de la función `console.log(argumento)`
- Los lenguajes de programación como JavaScript poseen operaciones aritméticas básicas como:
  - (+) suma
  - (-) resta
  - (\*) multiplicación
  - (/) división
  - (\*\*\*) potencia

# ALGUNOS EJEMPLOS: SUMA

```
let num1 = 5;  
let num2 = 10;  
let suma = num1 + num2;  
console.log(suma);
```

¿Qué imprimirá el `console.log`?

Rta: 15

# ALGUNOS EJEMPLOS: DIVISIÓN

```
let num1 = 10;  
let num2 = 5;  
let cociente = num1 / num2;  
console.log(cociente);
```

¿Qué imprimirá el **console.log**?

Rta: 2

# ALGUNOS EJEMPLOS: POTENCIA

```
let base = 2;  
let exponente = 3;  
let resultado = base ** exponente;  
console.log(resultado);
```

¿Qué imprimirá el `console.log`?

Rta: 8

# PROGRAMA

1. ENTORNO DE PROGRAMACIÓN
2. PROGRAMAR EN EL EDITOR
3. VARIABLES, CONSTANTES Y SUS TIPOS
4. OPERACIONES BÁSICAS
5. CONDICIONALES
6. CICLOS
7. FUNCIONES

2.

# PROGRAMAR EN EL EDITOR



# INSTALANDO JAVASCRIPT

- **Node.js** es un entorno de tiempo de ejecución de JavaScript multiplataforma y de código abierto.
- Se instala en cualquier sistema operativo (windows, linux o macos) desde:

<https://nodejs.org/es>

# EDITORES E IDES

- [Visual Studio Code](#)
- [Sublime Text](#)
- [Atom](#)
- [Visual Studio](#)
- [Notepad++](#)
- [Webstorm](#)
- [Eclipse](#)

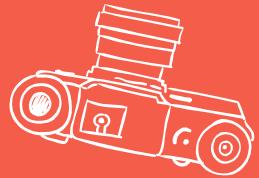
# HELLO WORLD!

1. Abre un editor de código
2. Crea un archivo nuevo y guárdalo con el nombre "hello-world.js".
3. Escribe el siguiente código en el archivo:
  - `console.log("Hello World!");`
4. Guarda el archivo y cierra el editor de código.
5. Abre la terminal(\*) y navega hasta la carpeta donde guardaste el archivo "hello-world.js".
6. Ejecuta el siguiente comando: **node hello-world.js**
7. Deberías ver la siguiente salida en la consola: **Hello, World!**

# (\*) ¿CÓMO ABRIR UNA TERMINAL?

- En Windows:
  1. Presiona la tecla de Windows en tu teclado o haz clic en el botón de Inicio de Windows en la barra de tareas.
  2. Escribe "cmd" en el cuadro de búsqueda y presiona Enter. Se abrirá la terminal de Windows.
- En macOS:
  1. Haz clic en el ícono de Spotlight en la esquina superior derecha de la pantalla o presiona la combinación de teclas "cmd + espacio" en tu teclado.
  2. Escribe "terminal" en la barra de búsqueda y presiona Enter. Se abrirá la terminal de macOS.
- En Linux:
  1. Presiona la combinación de teclas "Ctrl + Alt + T" en tu teclado. Se abrirá la terminal de Ubuntu.

# PROGRAMACIÓN IMPERATIVA EN JAVASCRIPT



# PROGRAMA

1. ENTORNO DE PROGRAMACIÓN
2. PROGRAMAR EN EL EDITOR
3. VARIABLES, CONSTANTES Y SUS TIPOS
4. OPERACIONES BÁSICAS
5. CONDICIONALES
6. CICLOS
7. FUNCIONES

# REPASO DE LA CLASE ANTERIOR

1. Formalidades del curso
2. ¿Qué es un programa? ¿Qué es programar? ¿Qué es JavaScript?
3. Compilado vs. Interpretado
4. Ejemplo: HTML + CSS + JavaScript
5. Ejemplos: suma, división y potencia
6. Instalar JavaScript (node) y algún editor (vscode)
7. Programamos y ejecutamos un **HELLO WORLD!**

# GIT

- Git es un sistema de control de versiones distribuido gratuito y de código abierto diseñado para manejar todo, desde proyectos pequeños hasta proyectos muy grandes, con rapidez y eficiencia.
- Se instala en cualquier sistema operativo (windows, linux o macos) desde: <https://git-scm.com/downloads>
- Existen varios proveedores, los más conocidos: [github](#), [bitbucket](#) y [gitlab](#). Usaremos [github](#).

# COMANDOS BÁSICOS DE GIT

ESTOS SON LOS QUE USAMOS SOLO LA PRIMERA VEZ

- **git init:** Inicializa un repositorio git en el directorio actual
- **git remote:** Agrega un repositorio remoto a la configuración del repositorio local

# COMANDOS BÁSICOS DE GIT

ESTOS SON LOS QUE VAMOS A USAR SIEMPRE

- **git status:** Muestra el estado actual de los archivos en el directorio de trabajo y en el índice
- **git add [nombre del archivo]:** Agrega un archivo al índice (también conocido como el área de preparación) para ser confirmado
- **git commit -m "Mensaje de confirmación":** Confirma los cambios agregados al índice con un mensaje de confirmación descriptivo
- **git push origin main:** Envía los cambios confirmados a un repositorio remoto

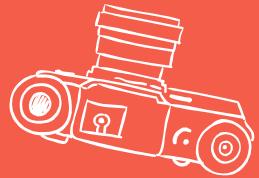
# COMANDOS BÁSICOS DE GIT

ESTOS SON LOS VAMOS A USAR CON MENOS FRECUENCIA

- **git diff:** Muestra las diferencias entre el directorio de trabajo y el índice
- **git pull origin main:** Obtiene los cambios de un repositorio remoto y los fusiona con la rama actual
- **git log:** Muestra el historial de confirmaciones

# PRACTIQUEMOS CON GIT EN VIVO

# PROGRAMACIÓN IMPERATIVA EN JAVASCRIPT



# PROGRAMA

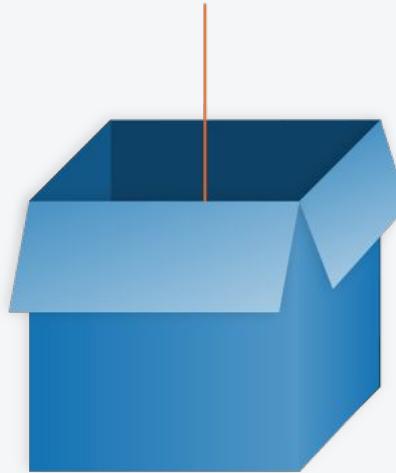
1. ENTORNO DE PROGRAMACIÓN
2. PROGRAMAR EN EL EDITOR
3. VARIABLES, CONSTANTES Y SUS TIPOS
4. OPERACIONES BÁSICAS
5. CONDICIONALES
6. CICLOS
7. FUNCIONES

# 3.

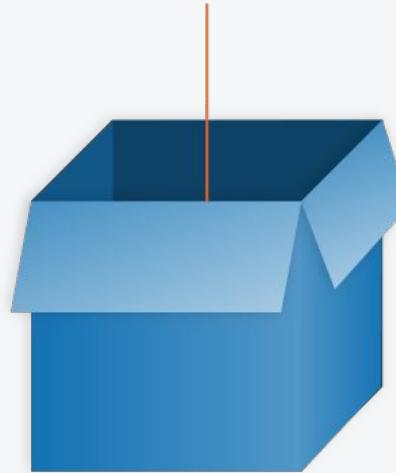
## VARIABLES, CONSTANTES Y SUS TIPOS

# ¿QUÉ ES UNA VARIABLE?

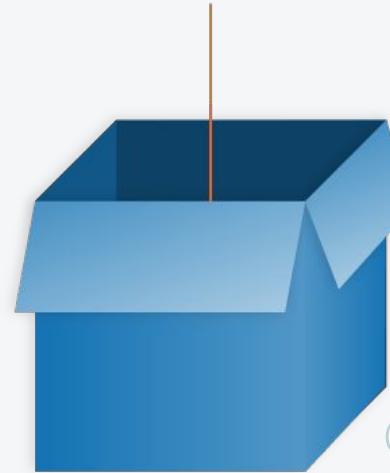
“Bob”



true



35



LIVE CODE

# EJEMPLO

```
let num1 = 10;  
num1 = 5;  
num1 = 1234567;
```

Veamos el ejemplo en el intérprete del navegador

# ¿POR QUÉ SON IMPORTANTES LAS VARIABLES?

## USANDO UNA VARIABLE

```
let name = prompt('¿Cuál es tu nombre?');

alert('¡Hola ' + name + ', encantado de verte!');
```

## SIN USAR VARIABLES

```
let name = prompt('¿Cuál es tu nombre?');

if (name === 'Adam') {
    alert('¡Hola Adam, encantado de verte!');
} else if (name === 'Alan') {
    alert('¡Hola Alan, encantado de verte!');
} else if (name === 'Bella') {
    alert('¡Hola Bella, encantado de verte!');
} else if (name === 'Bianca') {
    alert('¡Hola Bianca, encantado de verte!');
} else if (name === 'Chris') {
    alert('¡Hola Chris, encantado de verte!');
}
// ... y así sucesivamente
```

# DECLARAR UNA VARIABLE: VAR O LET

```
// Declarar la variable
let myName;
var myAge;

// Revisar su valor
myName;
myAge;

// Variable no declarada, error
scoobyDoo;

// Diferencias entre var y let
let myName = 'Chris';
let myName = 'Bob';

// Deberia reescribir como a continuación
let myName = 'Chris';
myName = 'Bob';
```

# INICIAR UN VARIABLE

```
// Inicializando variables ya declaradas  
myName = 'Chris';  
myAge = 37;  
  
// Revisar su valor  
myName; // 'Cris'  
myAge; // 37  
  
// Declarar e inicializar en un mismo paso  
let myDog = 'Rover';
```

# ACTUALIZAR UNA VARIABLE

```
// Definición y asignación  
let myName = 'Chris';  
let myAge = 37;
```

```
// Actualización de valores  
myName = 'Bob';  
myAge = 40;
```

# NOMENCLATURA DE VARIABLES

1. Utilizar sólo los caracteres: 0-9, a-z, A-Z
2. No use guiones bajos al comienzo
3. No uses números al comienzo
4. Seguir la convención **camelCase**
5. Nombres intuitivos, que describen los datos que contienen
6. Las variables distinguen entre mayúsculas y minúsculas
7. No puedes usar palabras como var, function, let y for

# NOMBRES DE VARIABLES: BUENOS Y MALOS

MALOS

```
1  
a  
_12  
myage  
MYAGE  
var  
Document  
skjfnndskjfbdskjfb  
thisisareallylongstupidvariablenameman
```

BUENOS

```
age  
myAge  
init  
initialColor  
finalOutputValue  
audio1  
audio2
```

# TIPOS DE VARIABLES

- **number:** 3.14, -4.12                          => let pi = 3.14...;
- **string:** “hola mundo”                          => let goodbye = 'Hasta luego';
- **booleanos:** true/false                          => let iAmAlive = true;

# TIPOS DE VARIABLES

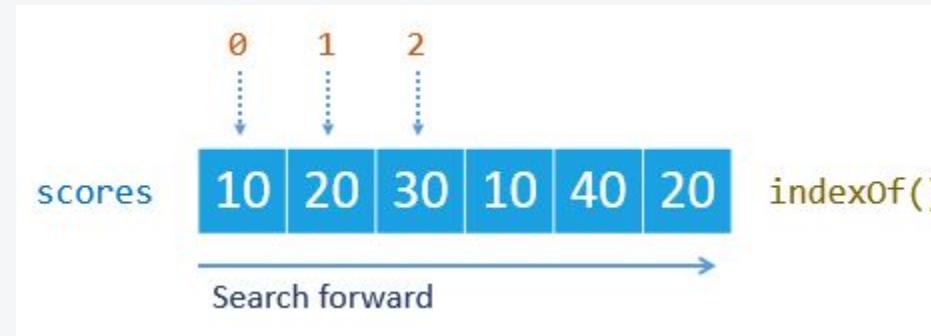
- **arreglos:**

- let miArregloDeNombres = ['Mariano', 'Emilia', 'Valentina'];
- let miArregloDeNumeros = [10, 15, 40];

- **objetos:**

- let perro = {nombre : 'Toby', raza : 'Dalmata' };
- let persona = {nombre: 'Agustín', edad : 27}

# ¿ CÓMO ACCEDER A LOS ÍNDICES DE UN ARREGLO ?



- largo del arreglo: 6
- `scores[2] = 30`
- indices: 0, 1, 2, 3, 4 y 5
- valores: 10, 20, 30, 10, 40 y 20

# ¿ CÓMO ACCEDER A LOS ÍNDICES DE UN ARREGLO ?

## Ejemplos:

- let miArregloDeNombres = ['Mariano', 'Emilia', 'Valentina'];
  - miArregloDeNombres[1] = 'Emilia'
- let miArregloDeNumeros = [10, 15, 40];
  - miArregloDeNumeros[0] = 10

# ¿ CÓMO ACCEDER A LOS ATRIBUTOS DE UN OBJETO ?

## Ejemplos:

- let perro = {nombre : 'Toby', raza : 'Dalmata' };
  - perro.nombre = 'Toby'
- let persona = {nombre: 'Agustín', edad : 27}
  - persona.edad = 27

# TIPADO DINÁMICO

```
let myString = 'Hello';

// incluso si el valor contiene números
let myNumber = '500'; // esto sigue siendo una cadena
typeof myNumber;

myNumber = 500; // mucho mejor -- ahora este es un número
typeof myNumber;
```

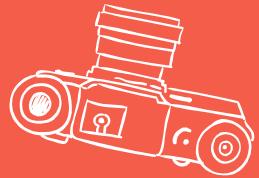
# CONSTANTES EN JAVASCRIPT

```
const daysInWeek = 7;  
const hoursInDay = 24;  
  
// si pruebo cambiar, me dará error  
daysInWeek = 8;
```

# PONGAMOS A PRUEBA LO APRENDIDO

## Ejercicios con variables

# PROGRAMACIÓN IMPERATIVA EN JAVASCRIPT



# PROGRAMA

1. ENTORNO DE PROGRAMACIÓN
2. PROGRAMAR EN EL EDITOR
3. VARIABLES, CONSTANTES Y SUS TIPOS
4. OPERACIONES BÁSICAS
5. CONDICIONALES
6. CICLOS
7. FUNCIONES

# REPASO DE LA CLASE ANTERIOR

1. ¿Qué es una variable? ¿Por qué son importantes?
2. Diferencias entre var y let
3. Cómo inicializar y actualizar una variable
4. Nomenclatura de nombres de variables
5. Tipos de variables: number, strings, booleanos, arreglos y objetos
6. Cómo acceder a los índices de un arreglo
7. Cómo acceder a los atributos de un objeto
8. Tipado dinámico en JavaScript
9. Constantes en JavaScript

# ENTREGA PRÁCTICO NO

- fecha límite: jueves 1 de junio, 22hrs.
- via github con un tag en el repo
  - git tag practicoo
  - git push origin main --tags
- las preguntas del práctico en un file respuestas.txt

# GLOSARIO

<https://developer.mozilla.org/es/docs/Glossary>

# TIPOS DE VARIABLES

- **number:** 3.14, -4.12 => let pi = 3.14...;
  - **string:** "hola mundo" => let goodbye = 'Hasta luego';
  - **booleanos:** true/false => let iAmAlive = true;

# TIPOS DE VARIABLES

- **undefined**      =>      let myVar;
- **null**                =>      myVar = null;
- **symbol**             =>      let mySym = Symbol();

## TIPOS DE VARIABLES: SYMBOL

- `console.log(Symbol(1) === Symbol(1));`
- `console.log(Symbol('foo') === Symbol('foo'));`

# TIPOS DE VARIABLES: SYMBOL

SIN SYMBOL

```
let people = [];
people['bob'] = { name: 'Bob Smith' };
people['bob'] = { name: 'Bob Jones' };
```

CON SYMBOL

```
let symbolPeople = [];
symbolPeople[Symbol('bob')] = { name: 'Bob Smith' };
symbolPeople[Symbol('bob')] = { name: 'Bob Jones' };
```

# PENDIENTES DE LA CLASE PASADA: ARREGLOS

- colección ordenada
  - `let myArray = [3, 2, 7, 9];`
- tipos de datos mixtos
  - `let myMixedArray = [0, 2.3, "hello", {foo: "bar"}];`
- propiedad **length**
  - `myArray.length;` => 4
  - `myMixedArray.length;` => 4

# PENDIENTES DE LA CLASE PASADA: ARREGLOS

- tamaño dinámico
  - `myMixedArray[4] = "bye";`
- indexación basada en cero
  - primer elemento: `myArray[0];`
  - ultimo elemento: `myArray[myArray.length - 1];`
- mutabilidad
  - `myMixedArray[0] = 3.44;`