**II\_10\_LENGUAJES Y PARADIGMAS DE PROGRAMACION**

SER MUY ESPECÍFICOS a través de los algoritmos.

LENGUAJES FORMALES: Lenguajes de programaicón

**TIPOS DE LENGUAJES**

ESPECÍFICOS: ejemplos los matemáticos

GENERALES: infinidad de aplicaciones

ALTO NIVEL: más cercanos al lenguaje natural. Ejemplo javascript

BAJO NIVEL: eficiencia en uso de recursos

Los lenguajes de programación LP Cambian con el tiempo.

**TIPADO**

Los lenguajes tipados fuerte y débil se distinguen según si permiten o no violaciones de los tipos de datos una vez declarados.

**TIPADO DÉBIL:**  no se indica, la mayoría de las veces, el tipo de variable. podemos operar con variables de distintos tipos.

Su principal ventaja es que es mucho más rápido de desarrollar, pero una clara desventaja es que podemos cometer muchos más errores si no tenemos cuidado.

Ejemlos: javascript, perl, lisp, swiprolog, php

**TIPADO FUERTE:** En estos lenguajes se nos obliga a indicar el tipo de dato al declarar la variable. Además, dicho tipo no puede ser cambiado una vez definida la variable. La **ventaja** es que al ser código más expresivo, cometeremos menos errores. La **desventaja** es que son mucho más estrictos a la hora de programar y que hay que escribir mucho más código.

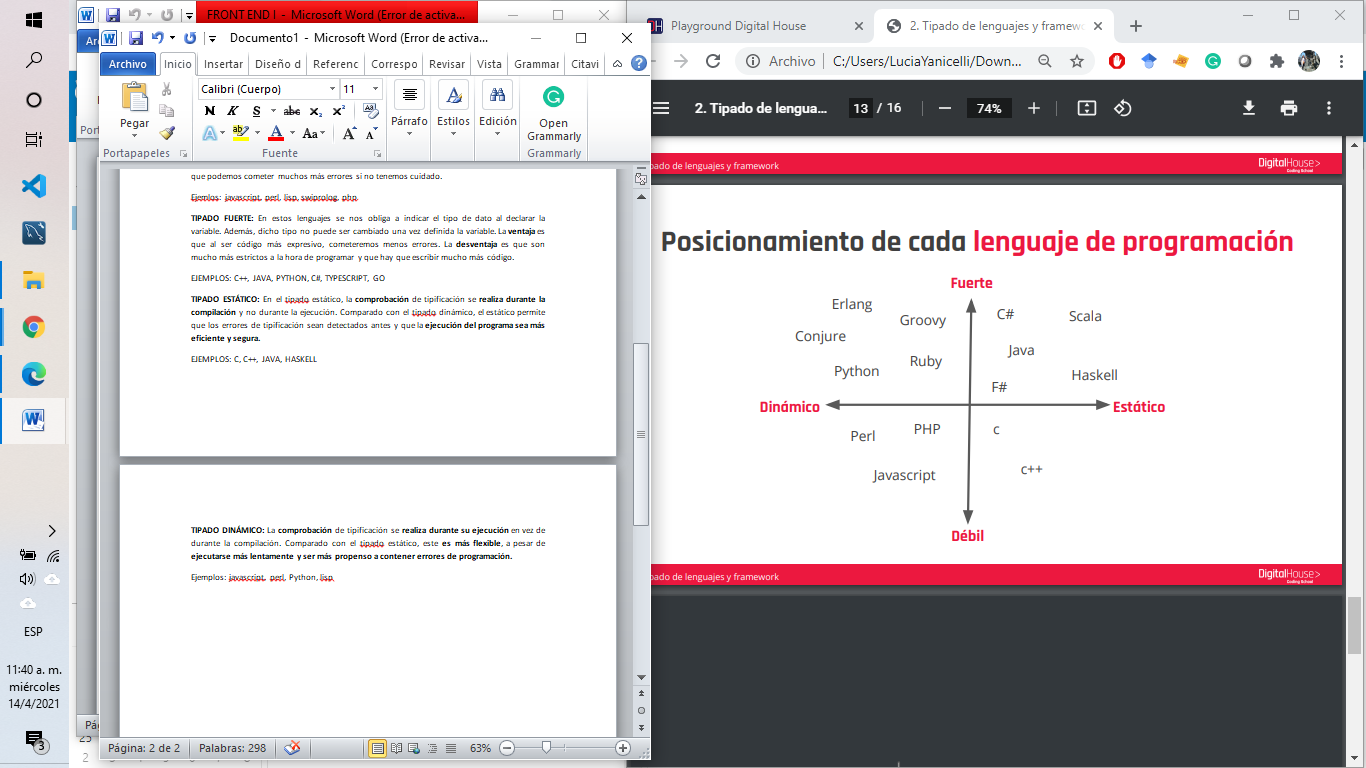
EJEMPLOS: C++, JAVA, PYTHON, C#, TYPESCRIPT, GO

**TIPADO ESTÁTICO:** En el tipado estático, la **comprobación** de tipificación se **realiza durante la compilación** y no durante la ejecución. Comparado con el tipado dinámico, el estático permite que los errores de tipificación sean detectados antes y que la **ejecución del programa sea más eficiente y segura.**

EJEMPLOS: C, C++, JAVA, HASKELL

**TIPADO DINÁMICO:** La **comprobación** de tipificación se **realiza durante su ejecución** en vez de durante la compilación. Comparado con el tipado estático, este **es más flexible**, a pesar de **ejecutarse más lentamente** **y ser más propenso a contener errores de programación.**

Ejemplos: javascript, perl, Python, lisp



**Frameworks** ~ Marco de Trabajo Es una estructura previa / esqueleto que se puede aprovechar para desarrollar un proyecto. El Framework es una especie de plantilla, un esquema conceptual, que simplifica la elaboración de una tarea, ya que solo es necesario complementarlo de acuerdo a lo que se quiere realiza

QUIZ

[1](https://playground.digitalhouse.com/)

**[¿Cuál es el lenguaje más cercano al lenguaje natural?](https://playground.digitalhouse.com/)**

Lenguaje de alto nivel.

**Corrección**   ¡Buen trabajo!

[2](https://playground.digitalhouse.com/)

**[Para aprender a programar es necesario aprender todos los lenguajes posibles.](https://playground.digitalhouse.com/)**

Falso.

**Corrección**  ¡Buen trabajo!

[3](https://playground.digitalhouse.com/)

**[Los lenguajes de tipado débil:](https://playground.digitalhouse.com/)**

Nos permite desarrollar mucho más rápido.

**Corrección**   ¡Buen trabajo!

[4](https://playground.digitalhouse.com/)

**[En los lenguaje de tipado dinámico, la comprobación de sobre la tipificación de variables se realiza durante su ejecución y no durante la compilación.](https://playground.digitalhouse.com/)**

Verdadero.

**PARADIGMA DE PROGRAMACIÓN**

La programación al inicio era estructurada: muy compleja

Cambio de perspectiva: cada objeto tenía un estado interno y funciones. LP: simula 67. Paradigma de programación ORIENTADO A OBJETOS.

PARADIGMA: forma de pensar en un lenguaje de programación

¿Qué es un paradigma?

Paradigmas de programación

Un paradigma es una forma de pensar bajo un modelo preestablecido.

**PARADIGMAS**

**Paradigma estructurado**

Sigue una línea de pensamiento donde se suele ejecutar una instrucción a la vez y uno se rige en un acotado set de instrucciones.

Este paradigma es muy utilizado para el desarrollo de sistemas.

**Paradigma de programación orientado a objetos**

El código puede agruparse de tal forma que llegue a representar una entidad y que interprete mensajes. La fortaleza del paradigma de la programación orientada a objetos yace en utilizar abstracciones y crear entidades.

**Paradigma funcional**

El paradigma de programación funcional se basa en un concepto muy simple y es el de las funciones matemáticas.

La fortaleza de este paradigma radica en que siempre que a la función X se le pasa el valor A, esta siempre va a devolver el valor B.

Esta propiedad de devolver el mismo valor se le conoce como inmutabilidad, y es característico de este paradigma.

**Paradigma lógico**

En lugar de desarrollar pasos e instrucciones, utiliza reglas lógicas para consultar al sistema y el mismo infiere que hacer en base a las reglas lógicas establecidas.

**Paradigma de programación con lenguaje específico de dominio**

Los lenguajes que encontramos acá tratan de resolver problemáticas superespecíficas.

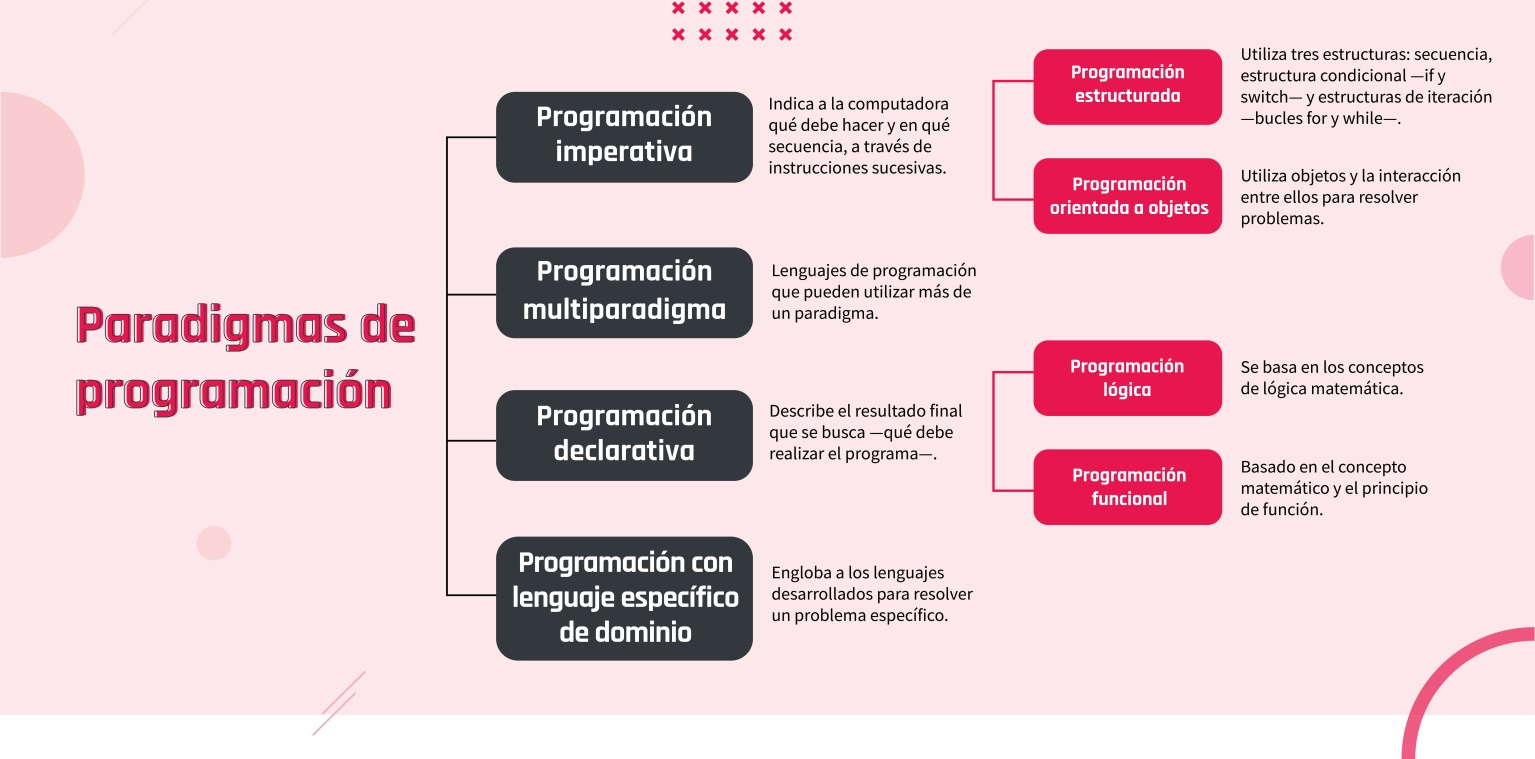
Ejemplo:

Cuando queremos consultar una base de datos de un supermercado para saber qué productos tenemos en la categoría de electrodomésticos.

**Multiparadigma**

A lo largo de la evolución de la programación, con nuevos desafíos y paradigmas han habido lenguajes que han modificado su estructura para poder permitir dar soluciones en distintos paradigmas

En JavaScript se puede escribir código tanto con el paradigma estructurado como con programación orientada a objetos e incluso utilizar el paradigma funcional.



QUIZ

[1](https://playground.digitalhouse.com/)

#### [Un paradigma es:](https://playground.digitalhouse.com/)

Una forma de pensar bajo un modelo preestablecido.

**Corrección**  ¡Buen trabajo!

[2](https://playground.digitalhouse.com/)

#### [¿Qué paradigma describe el resultado final?](https://playground.digitalhouse.com/)

Programación declarativa.

**Corrección**  ¡Buen trabajo!

[3](https://playground.digitalhouse.com/)

#### [Los lenguajes que encontramos en este paradigma tratan de resolver problemáticas específicas](https://playground.digitalhouse.com/)

Programación con lenguaje específico de dominio.

**Corrección**   ¡Buen trabajo!

[4](https://playground.digitalhouse.com/)

#### [En el paradigma de programación orientada a objetos el código puede agruparse de tal forma que llegue a representar entidades que interactúen entre sí a través de mensajes.](https://playground.digitalhouse.com/)

Verdadero.

**Corrección**   ¡Buen trabajo!

**DEL CÓDIGO AL EJECUTABLE**

**QUIZ**

[1](https://playground.digitalhouse.com/)

#### [¿Qué hace la compilación?](https://playground.digitalhouse.com/)

Toma el código fuente y lo transforma en un programa ejecutable escrito en código máquina.

**Corrección**  ¡Buen trabajo!

[2](https://playground.digitalhouse.com/)

#### [¿Qué podemos utilizar para que nuestro código sea independiente de la plataforma donde corra?](https://playground.digitalhouse.com/)

Podemos utilizar máquinas virtuales o intérpretes.

**Corrección**  ¡Buen trabajo!

[3](https://playground.digitalhouse.com/)

**[Un intérprete analiza línea por línea un código fuente en el momento de su ejecución.](https://playground.digitalhouse.com/)**

Verdadero.

**Corrección**  ¡Buen trabajo!

[4](https://playground.digitalhouse.com/)

#### [La principal ventaja de compilar un código en una máquina virtual, es la alta performance.](https://playground.digitalhouse.com/)

Falso.