GENERALITAT VALENCIANA
Conselleria d'Educació, Cultura i Esport

ceedcv CENTRE ESPECÍFIC D'EDUCACIÓ A DISTÀNCIA DE LA COMUNITAT VALENCIANA

Unidad 1. Desarrollo de software

Conceptos básicos y lenguajes de programación

Raúl Palao Lozano
18/09/2023

1.1 Conceptos básicos

1.1. CONCEPTOS BÁSICOS

- La palabra **informática** viene de “información” y “automática”.

  Información: un conjunto de datos

  Automático: que funciona por sí mismo

¿Qué es un dato?

- Así pues, la informática es el proceso de la información y su tratamiento, mediante un sistema automático, como puede ser un “*ordenador*”.

¿Te parece correcto el término ordenador?

- Toda la información que usa un dispositivo debe estar representada en una secuencia de ceros y unos (**dígitos**) llamada **código binario**.

- Cuando un dispositivo cualquiera maneja o almacena información en forma de bits (dígitos) decimos que ese dispositivo es **digital**.

¿De dónde viene la palabra dígito?

1.1. CONCEPTOS BÁSICOS

- La palabra **informática** viene de “información” y “automática”.

  Información: un conjunto de datos

  Automático: que funciona por sí mismo

¿Qué es un dato?

- Así pues, la informática es el proceso de la información y su tratamiento, mediante un sistema automático, como puede ser un “*ordenador*”.

¿Te parece correcto el término ordenador?

- Toda la información que usa un dispositivo debe estar representada en una secuencia de ceros y unos (**dígitos**) llamada **código binario**.

- Cuando un dispositivo cualquiera maneja o almacena información en forma de bits (dígitos) decimos que ese dispositivo es **digital**.

¿De dónde viene la palabra dígito?

1.1. CONCEPTOS BÁSICOS

- La palabra **informática** viene de “información” y “automática”.

  Información: un conjunto de datos

  Automático: que funciona por sí mismo

¿Qué es un dato? **Dato es la unidad mínima de información.**

- Así pues, la informática es el proceso de la información y su tratamiento, mediante un sistema automático, como puede ser un “*ordenador*”.

¿Te parece correcto el término ordenador? **Es más correcto decir dispositivo.**

- Toda la información que usa un dispositivo debe estar representada en una secuencia de ceros y unos (**dígitos**) llamada **código binario**.

- Cuando un dispositivo cualquiera maneja o almacena información en forma de bits (dígitos) decimos que ese dispositivo es **digital**.

¿De dónde viene la palabra dígito? **Digital viene del latín digitum (dedo), ya que podemos contar con ellos.**

1.1. CONCEPTOS BÁSICOS

- Es importante usar la palabra “**dispositivo**”, ya que hoy en día podemos programar desde un ordenador hasta una nevera, pasando por unas zapatillas deportivas o, por qué no, hasta una alfombra si estos manejan información digital.

- Cada vez más elementos de nuestra vida cotidiana aceptan órdenes en forma de comandos, líneas de código o programas, dentro de lo que se conoce como Internet de las Cosas, **Internet of Things** o con sus siglas en inglés IoT, un concepto que evoluciona del concepto clásico de domótica.

1.1. CONCEPTOS BÁSICOS

- El **software** es la parte intangible de un sistema informático, el equivalente al equipamiento lógico.

- Todo software está diseñado para realizar una tarea determinada en nuestro sistema y está presente en el sistema operativo (SO), en las aplicaciones que utilizamos y en, prácticamente, cualquier parte de un dispositivo electrónico moderno.

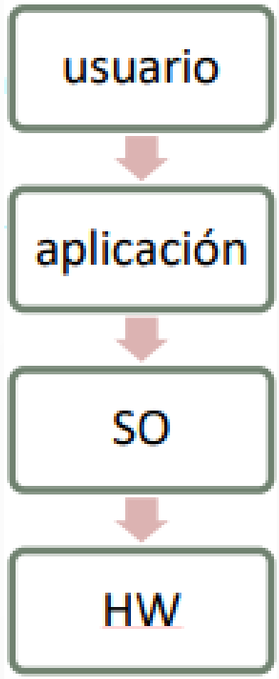
¿Conoces la diferencia entre HW y SO?

¿Y la relación entre SO, HW y SW?

1.1. CONCEPTOS BÁSICOS

- El **software** es la parte intangible de un sistema informático, el equivalente al equipamiento lógico.

- Todo software está diseñado para realizar una tarea determinada en nuestro sistema y está presente en el sistema operativo (SO), en las aplicaciones que utilizamos y en, prácticamente, cualquier parte de un dispositivo electrónico moderno.



¿Conoces la diferencia entre HW y SO?

¿Y la relación entre SO, HW y SW?

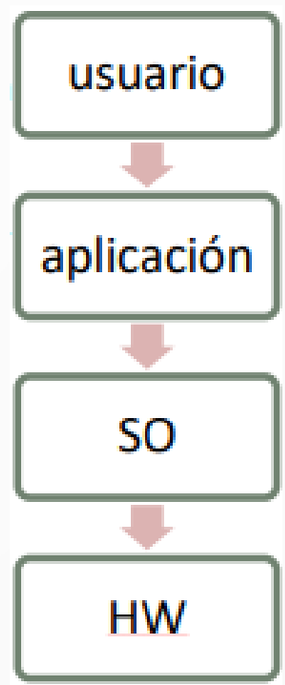
1.1. CONCEPTOS BÁSICOS

- El **software** es la parte intangible de un sistema informático, el equivalente al equipamiento lógico.

- Todo software está diseñado para realizar una tarea determinada en nuestro sistema y está presente en el sistema operativo (SO), en las aplicaciones que utilizamos y en, prácticamente, cualquier parte de un dispositivo electrónico moderno.

¿Conoces la diferencia entre HW y SO? **HW es lo que podemos tocar. SW es todo lo demás.**

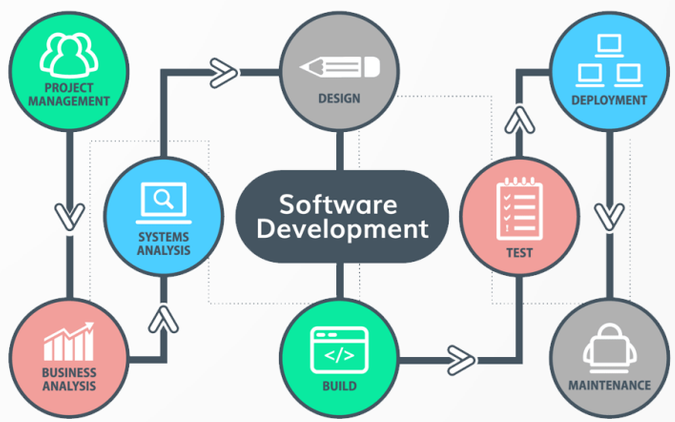
¿Y la relación entre SO, HW y SW? **El SO es un tipo de SW que nos ayuda a comunicarnos con el HW.**



1.1. CONCEPTOS BÁSICOS

- **Programar**, en esencia, consiste en darle órdenes y datos a un ordenador para recibir una serie de resultados o provocar cierto comportamiento.
- Para ser técnicamente correctos, usaremos codificar **instrucciones** en lugar de dar órdenes.

PROGRAMAR = ~~DAR ÓRDENES~~ = CODIFICAR INSTRUCCIONES



- **Desarrollar software** es o, al menos, debería ser mucho más que ~~darle órdenes~~ codificar instrucciones para un dispositivo, mucho más que programar.
- De hecho, el tiempo que dedicamos o que deberíamos dedicar, **a analizar, diseñar, probar, documentar y mantener** el software es mucho mayor (o debería serlo) que el tiempo que dedicamos exclusivamente a codificar instrucciones.

DESARROLLAR SW = ~~PROGRAMAR~~ = ANALIZAR, DISEÑAR, CODIFICAR, PROBAR, DOCUMENTAR, MANTENER

- Por tanto, diremos desarrollador de software (o simplemente **desarrollador**) en lugar de programador, si consideramos que nuestra labor se basa en “algo más” que codificar instrucciones.

1.1. CONCEPTOS BÁSICOS

- Un **programa** es una serie de ~~órdenes~~ instrucciones secuenciadas con una finalidad concreta y que devuelven un valor o realizan una función determinada.

  => Una función que dados números te devuelva la suma

  => Un botón que minimiza las ventanas de tu escritorio

- Una **librería** es un archivo, o contenedor lógico, que contiene una serie de programas.

  => Cualquiera de los archivos DLL que encuentras en la carpeta System32 de Windows son librerías

- Una **aplicación** está formada por uno o varios programas con (o sin) sus librerías correspondientes.
- => Adobe Photoshop es una aplicación (no un programa)

- Cuando lo que tenemos son varias aplicaciones que pueden ejecutarse independientemente, una de otra, suele denominarse **suite** o “paquete integrado”
- => MS Office o con Libre y Open Office son suites (no librerías, ni programas, ni aplicaciones)

1.1. CONCEPTOS BÁSICOS

- ¿Y el Sistema Operativo?

  => Windows, Linux, Mac ...

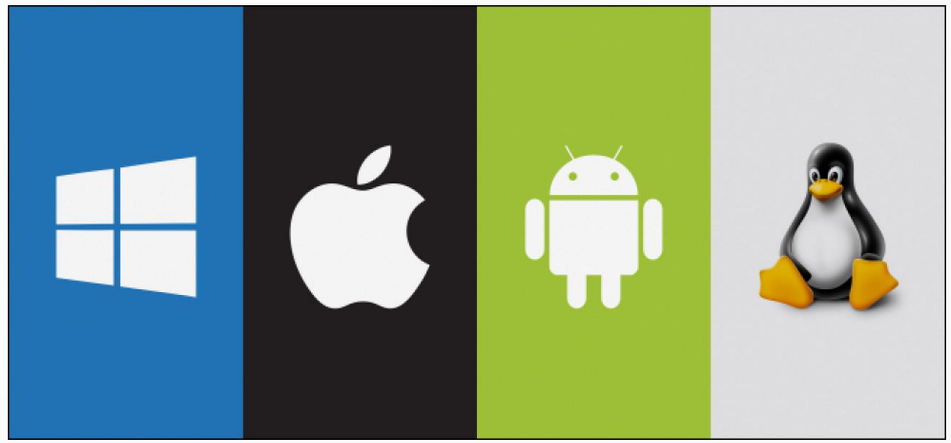
- ¿Se considera un programa, una librería, una aplicación, una suite?

- ¿Qué opinas?

1.1. CONCEPTOS BÁSICOS

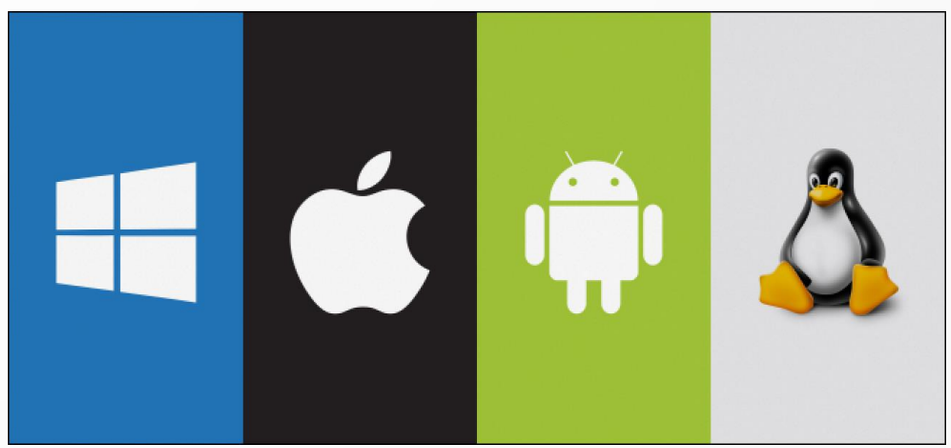
- ¿Se considera el sistema operativo un programa,

una librería, una aplicación, una suite?



1.1. CONCEPTOS BÁSICOS

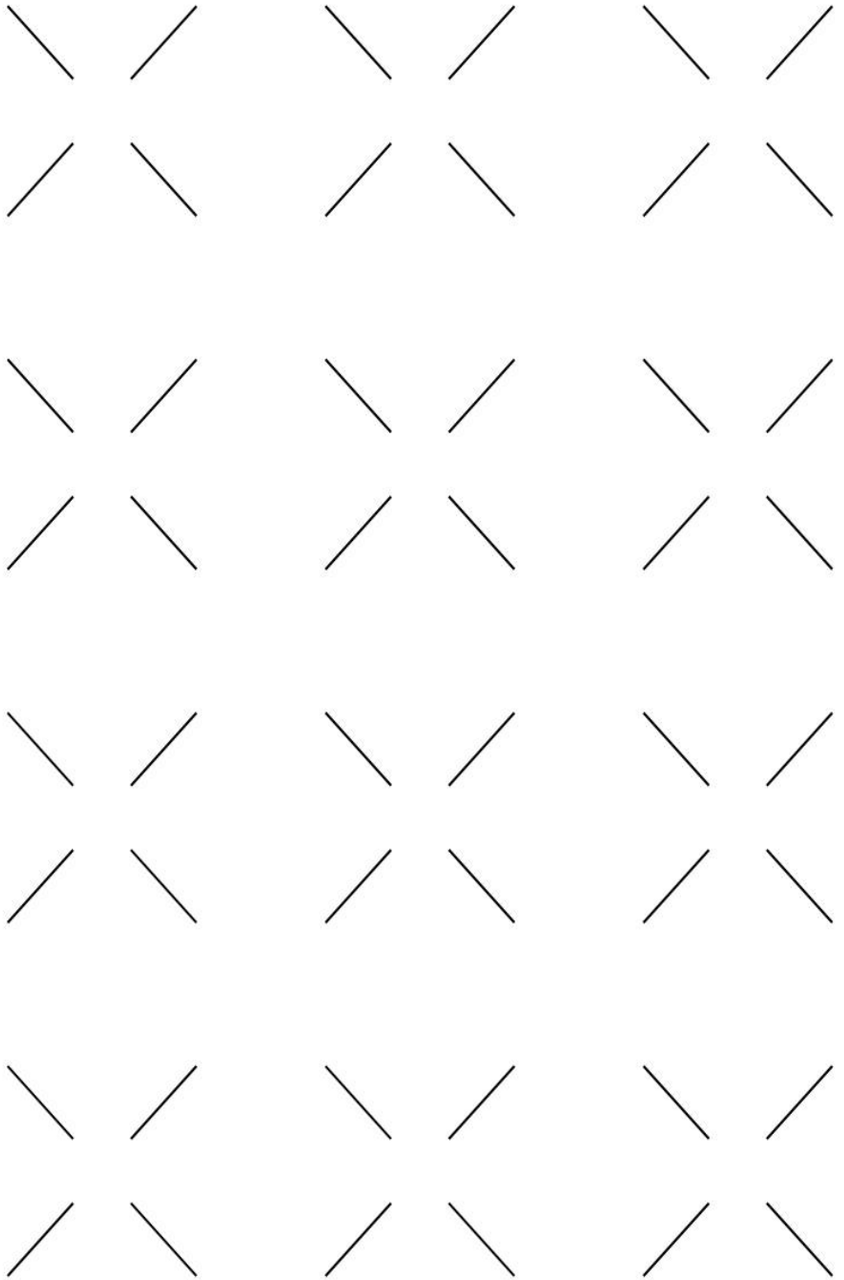
- ¿Se considera el sistema operativo un programa, una librería, una aplicación, una suite?



- => No hay una respuesta única pero quizás la más acertada sea que un SO está formado por una serie de programas, librerías, y aplicaciones.

- => Por tanto, se podría considerar un SO como una suite de aplicaciones que vienen ya preinstaladas y otras muchas que el usuario añade una vez toma el control.

1.2 Lenguajes de programación



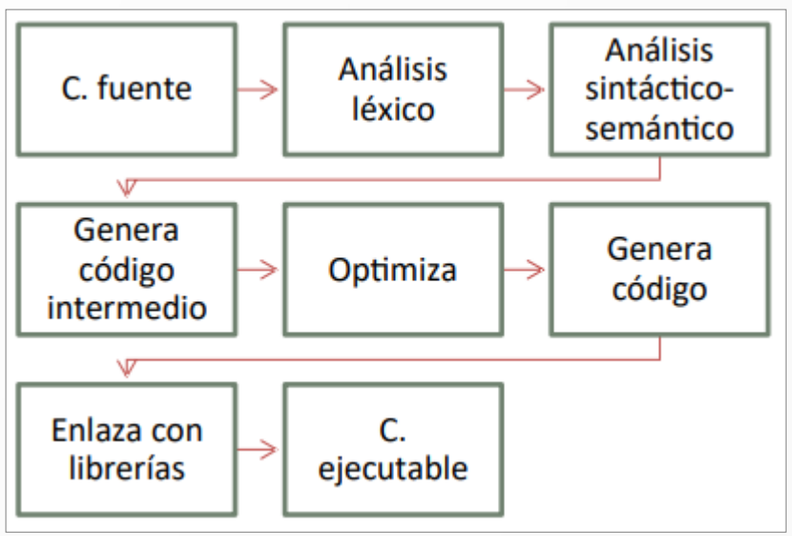
1.2. LENGUAJES DE PROGRAMACIÓN

- Al conjunto de TODAS las ~~órdenes~~ instrucciones que forman un programa le llamaremos código fuente.

- Piensa que, un mismo código fuente, para poder ser ejecutado en diferentes sistemas operativos necesita ser compilado de varias maneras.

- Por ejemplo, la calculadora de la que hablamos puede que tenga un único código fuente, pero seguro que tendrá diferentes ejecutables, uno por cada sistema operativo en el que quieras que funcione.

- Al código intermedio entre el fuente y el ejecutable se le llama código objeto.



1.2. LENGUAJES DE PROGRAMACIÓN

CLASIFICADOS POR EJECUCIÓN

Lenguajes compilados

=> Su código fuente es traducido por un compilador a un archivo ejecutable entendible para la máquina en determinada plataforma.

=> Con ese archivo se puede ejecutar el programa cuantas veces sea necesario sin tener que repetir el proceso por lo que el tiempo de espera entre ejecución y ejecución es ínfimo.

=> La familia C que incluye a C++, Objective C, C# y también otros como Fortran, Pascal, Haskell y Visual Basic.

=> Los vemos más en software de escritorio ya que requieren de mayores recursos y de acceso a archivos determinados.

=> También por el peso mayor que estos suelen tener en sus archivos ejecutables (**rápidos pero pesados**).

1.2. LENGUAJES DE PROGRAMACIÓN

CLASIFICADOS POR EJECUCIÓN

Lenguajes interpretados

=> No necesitan ser compilados. Se ~~ejecutan~~ interpretan línea por línea.

=> Ruby, Python, PHP, JavaScript y otros como Perl, Smalltalk, MATLAB, Mathematica.

=> Los lenguajes interpretados permiten el tipado dinámico de datos, es decir, no es necesario inicializar una variable con determinado tipo de dato sino que esta puede cambiar su tipo en condición al dato que almacene entre otras características más.

=> Los vemos más en software de entornos web o terminales de comandos, ya que requieren de menores recursos y de acceso a archivos determinados. (**más lentos pero ligeros**).

1.2. LENGUAJES DE PROGRAMACIÓN

CLASIFICADOS POR EJECUCIÓN

Lenguajes virtuales

=> Algunos autores clasifican a lenguajes como Java lenguajes virtuales o híbridos.

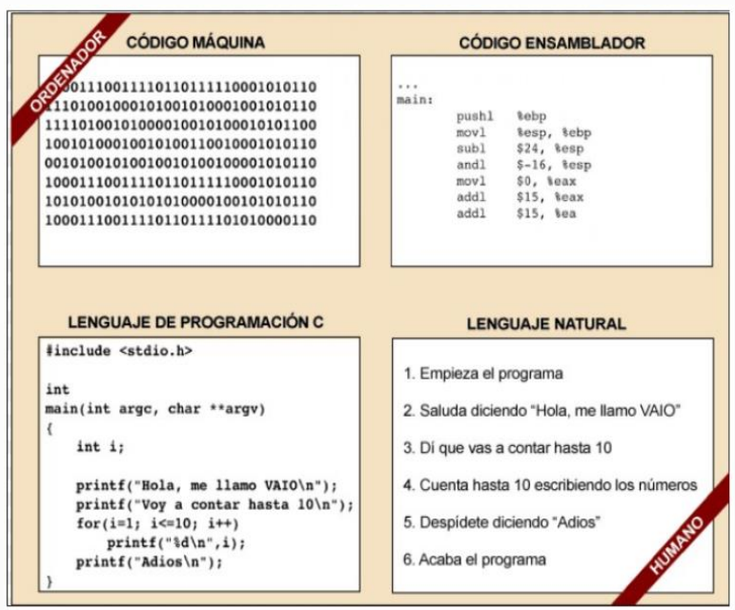
=> Es un caso particular ya que hace uso de una máquina virtual que se encarga de la traducción del código fuente por lo que hay veces es denominado compilado e interpretado o virtual. Otra

=> Otra ventaja de la máquina virtual que usar Java es que le permite ejecutar código Java en cualquier máquina que tenga instalada la JVM.

1.2. LENGUAJES DE PROGRAMACIÓN

CLASIFICADOS POR NIVEL DE ABSTRACCIÓN

- Respecto del **nivel de abstracción**, podría definirse como el nivel de cercanía al lenguaje natural, siendo un lenguaje de alto nivel el que se parece más al lenguaje natural y un lenguaje de bajo nivel el que es más similar al lenguaje máquina.



LENGUAJE DE ALTO NIVEL
(JAVA, C#, PHP, PYTHON, ETC)

LENGUAJE DE BAJO NIVEL
ENSAMBLADOR

CÓDIGO MÁQUINA
(BINARIO [0-1])

HARDWARE
(PARTE FISICA DE LA COMPUTADORA)

1.2. LENGUAJES DE PROGRAMACIÓN

CLASIFICADOS POR PARADIGMA

- Un **paradigma** de programación es un conjunto de teorías, estándares y métodos que juntos representan un medio de organización de conocimientos.

- Dicho con otras palabras, es un conjunto de características comunes que comparten varios lenguajes de programación.

- De ahí que existan lenguajes multiparadigma, al no encajar exactamente en ningún paradigma concreto y cumplir solo unas características de uno y de otro o, también se da el caso, de que cumpla todas las características de más de un paradigma.

1.2. LENGUAJES DE PROGRAMACIÓN

CLASIFICADOS POR PARADIGMA

- Los lenguajes **imperativos** dicen al dispositivo qué debe hacer y cómo hacerlo.

Imprime en pantalla en C++:

cout « "Hola mundo";

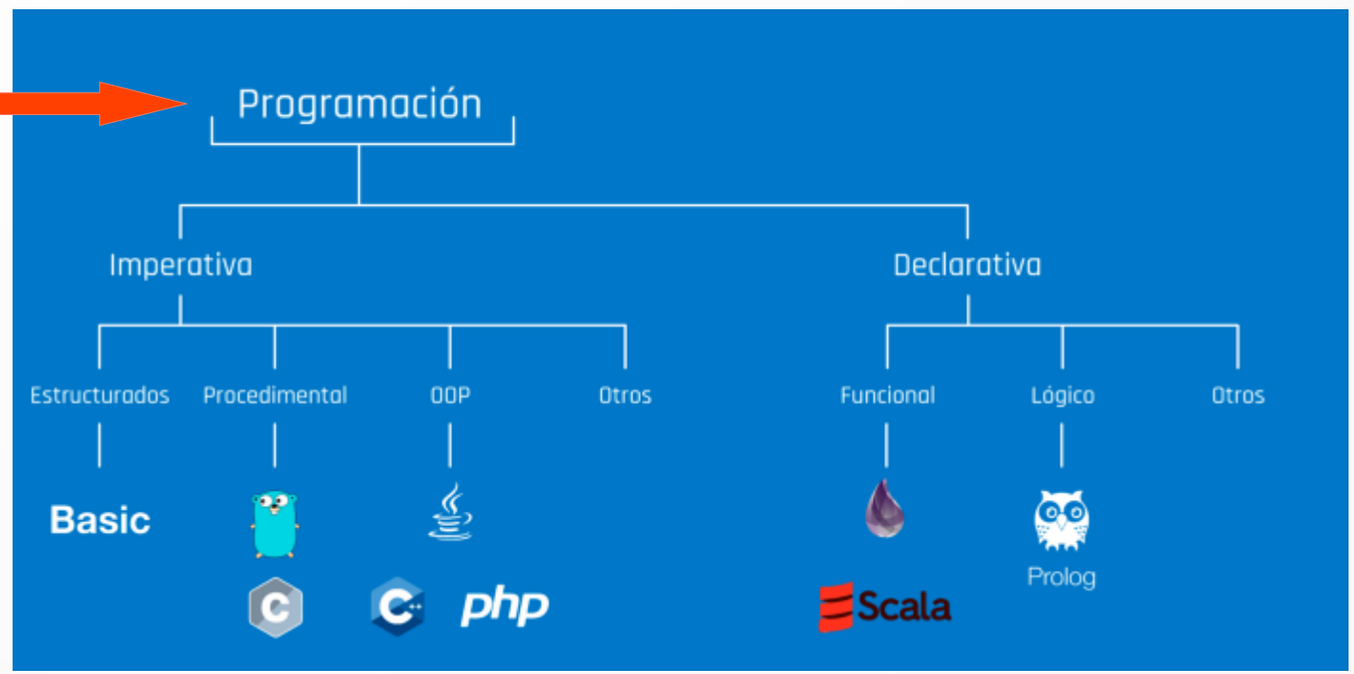
- Los lenguajes **declarativos** dicen al dispositivo qué debe hacer pero no cómo hacerlo.

*SELECT * FROM CLIENTES WHERE NOMBRE LIKE 'RAUL%';*

- En este ciclo veréis muchos lenguajes, prácticamente todos serán imperativos excepto SQL, que es declarativo.

1.2. LENGUAJES DE PROGRAMACIÓN

CLASIFICADOS POR PARADIGMA



// EDUCACIÓN A DISTANCIA // EDUCACIÓN PÚBLICA Y GRATUITA // EDUCACIÓN DE CALIDAD // EDUCACIÓN AUTONÓMICA // EDUCACIÓN A DISTANCIA // EDUCACIÓN PÚBLICA Y GRATUITA // EDUCACIÓN DE CALIDAD // EDUCACIÓN AUTONÓMICA // EDUCACIÓN A DISTANCIA // EDUCACIÓN PÚBLICA Y GRATUITA // EDUCACIÓN DE CALIDAD // EDUCACIÓN AUTONÓMICA // EDUCACIÓN A DISTANCIA // EDUCACIÓN PÚBLICA Y GRATUITA // EDUCACIÓN DE CALIDAD // EDUCACIÓN AUTONÓMICA // EDUCACIÓN A DISTANCIA // EDU

GENERALITAT VALENCIANA
Conselleria d'Educació, Cultura i Esport

ceedcv
CENTRE ESPECÍFIC D'EDUCACIÓ A DISTÀNCIA DE LA COMUNITAT VALENCIANA