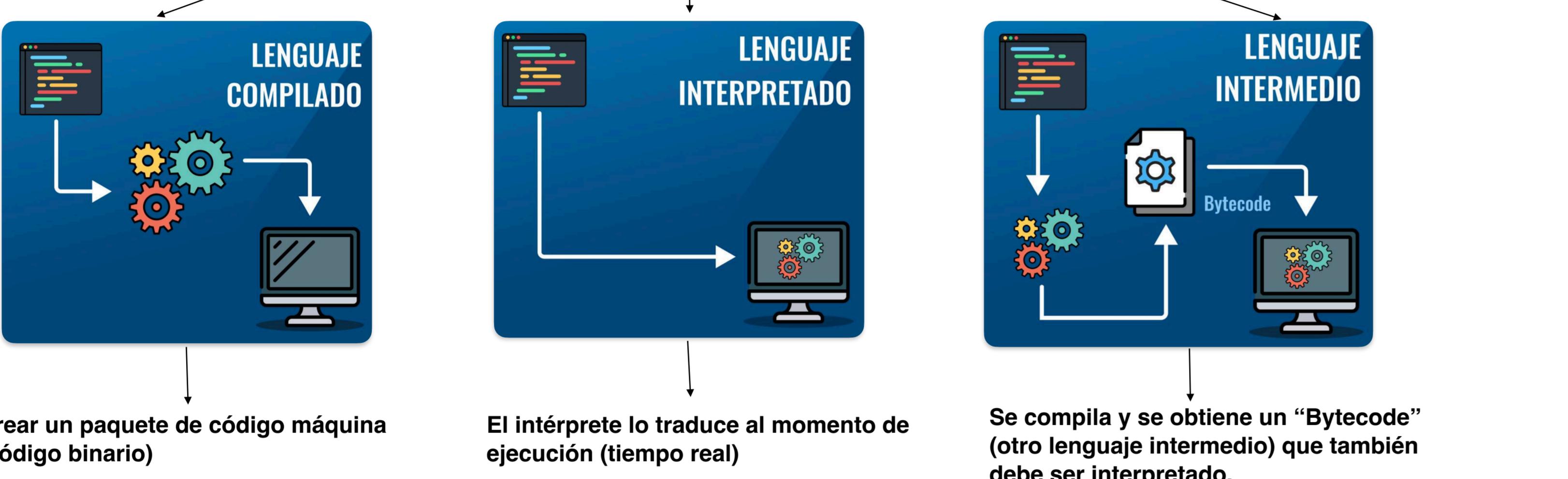


LENGUAJES DE PROGRAMACION



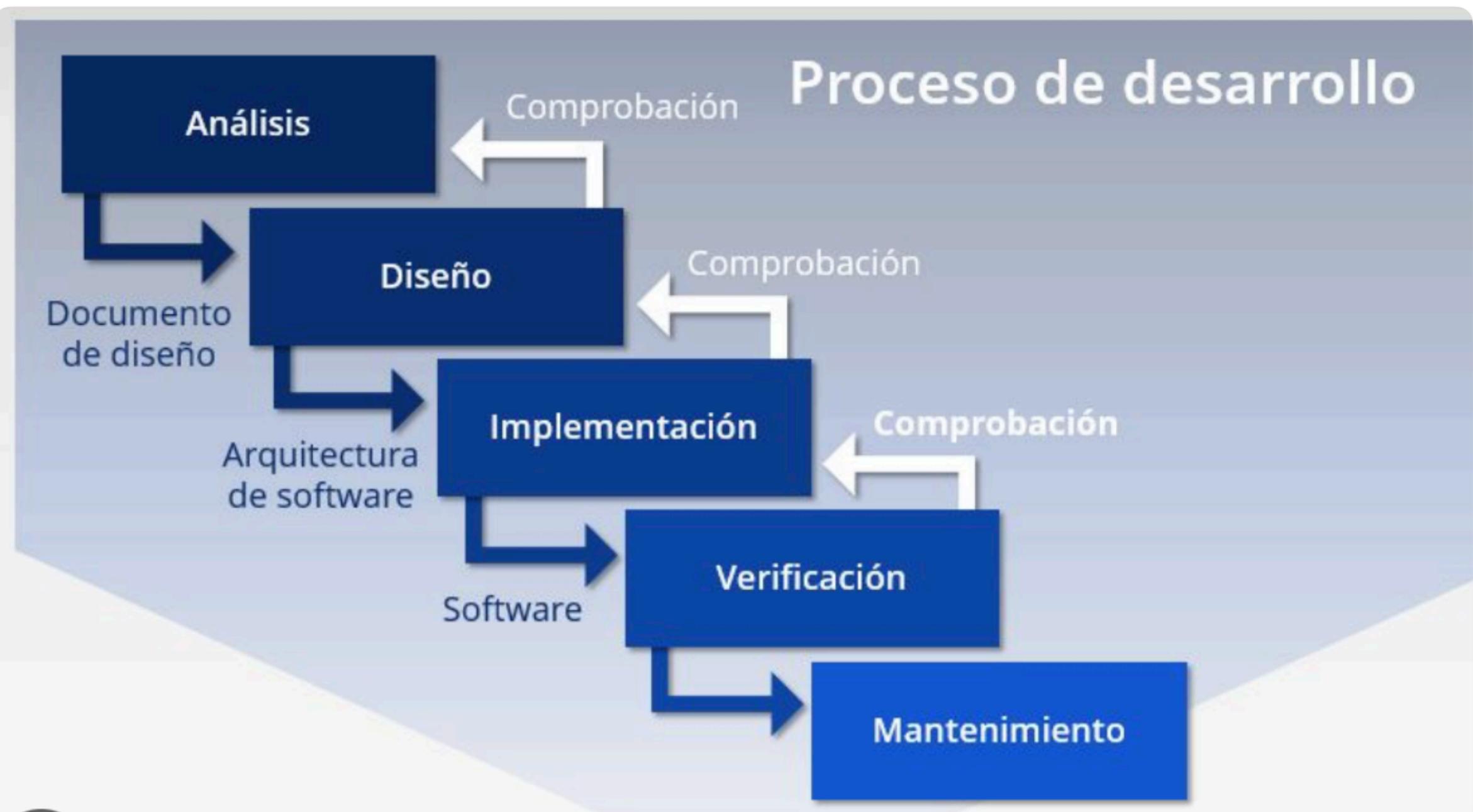
- De bajo nivel (cercaos a la máquina)
 - Lenguaje máquina
 - Lenguaje ensamblador (mnemotécnicos)
- De alto nivel (cercaos al usuario)
 - Lenguajes compilados (compilador) C, C++, Lisp, Java (a bytecode), etc.
 - Lenguajes interpretados (intérprete) Perl, PHP, Python, JavaScript, etc.

LENGUAJES DE PROGRAMACION MAS USADOS

Este es el ranking de los lenguajes de programación más usados en la actualidad, está basado en los datos del índice Tiobe el cual (hay que aclarar) no señala que lenguaje es mejor, sino, que lenguaje se escribió en **mayor cantidad** durante el último mes del 2022.

#1 Python: Es un lenguaje muy versátil ya que tiene múltiples usos de aplicación como Inteligencia Artificial, Big Data y desarrollo web, es de código abierto y muy fácil de aprender.	#2 Lenguaje C: Es uno de los primeros lenguajes que forma la base de otros como C++, C# o Java. Con él podemos desarrollar tanto aplicaciones como sistemas operativos.
#3 Java: Es un lenguaje de propósito general y su ámbito de aplicación es super amplio, es orientado a objetos y se utiliza en gran parte para crear aplicaciones y procesos en múltiples dispositivos.	#4 C++: Surgió como extensión de C para que pudiera manipular objetos, es muy utilizado en bases de datos, navegadores web, compiladores o videojuegos.
#5 C#: Es uno de los preferidos por ser uno de los más fáciles de aprender y para todo: aplicaciones web, servicios, aplicaciones móviles, no necesita compilación ninguna.	#6 Visual Basic: Es un lenguaje de programación de bajo nivel, creado por Microsoft similar a C pero orientado a objetos. Es muy usado en la industria de los juegos, robótica, impresión 3D, IoT y desarrollo móvil / web.
#7 JavaScript: Es uno de los más populares para ser uno de los más poderosos y flexibles que hay para todo: aplicaciones web, servicios, aplicaciones móviles, no necesita compilación ninguna.	#8 Assembly: Es un lenguaje de programación de bajo nivel. Consiste en un conjunto de instrucciones básicas para los computadores, microprocesadores y otros circuitos integrados programables.
#9 SQL: Es un lenguaje de programación utilizado para el manejo de información en las bases de datos relacionales, además, permite realizar operaciones de acceso y manipulación de la información.	#10 PHP: Es uno de los clásicos: es un lenguaje de uso general que se adopta especialmente al desarrollo web. PHP se encuentra instalado en más de 20 millones de sitios web.

Fases del desarrollo de software.



Análisis.

La fase de análisis define los requisitos del software que hay que desarrollar. Comenzaremos con una entrevista individual o una dinámica de grupo con los clientes. En ella también tendremos que saber lo que el cliente quiere o lo que cree que necesita.

Es importante que se mantenga una comunicación bilateral, y es necesario un consenso por ambas partes para llegar a definir los requisitos del software. Para ello se crea un informe ERS (Especificación de Requisitos del Sistema).

En esta fase, los requisitos que se deben definir son:

- **Requisitos funcionales:** Con estos requisitos describiremos detalladamente lo que realiza el sistema y como reacciona ante diferentes entradas y situaciones.
- **Requisitos no funcionales:** Con estos requisitos detallaremos por ejemplo la capacidad de almacenamiento o la fiabilidad del sistema. Estos requisitos no incluyen como funciona el sistema.

Diseño.

Determinaremos el funcionamiento del software de una forma global y general sin entrar en detalles. Uno de los objetivos principales es establecer las consideraciones de los recursos del sistema, tanto físicos como lógicos. Sin embargo, también se puede diseñar el sistema en función de los recursos de los que se dispone. En la fase de diseño se crean los diagramas de casos de uso y de secuencia para definir la funcionalidad del sistema.

Codificación.

En esta fase lo que de debe hacer es escribir el programa en el lenguaje de programación que consideremos más oportuno o en el que quiera el cliente en cuestión.

Pruebas.

Con una doble funcionalidad, las pruebas buscamos confirmar que el programa se ha escrito correctamente y el software no contiene errores. En esta fase realizaremos las siguientes pruebas.

- **Pruebas de verificación.** Se comprueba que el software se ha realizado correctamente.
- **Pruebas de validación.** Se comprueba si el producto se ajusta a los requisitos del cliente.

Documentación.

Cada etapa del desarrollo tiene que quedar perfectamente documentada. Teniendo en cuenta esto, la documentación, puede dividirse en dos clases.

1. **De proceso.** Se incluye el proceso de desarrollo de software y su mantenimiento.
2. **De producto.** Describe el software que se está desarrollando. Incluye:
 - **Documentación de usuario:**
 - Para usuarios finales. Se centra en como funciona la aplicación.
 - Para administradores del sistema. Gestionarán los programas que usan los usuarios finales
 - **Documentación del sistema:** Son los documentos que describen el sistema desde los requisitos hasta las pruebas de aceptación.

Explotación del sistema.

Una vez que tenemos nuestro software, hay que prepararlo para su distribución. Para ello se implementa el software en el sistema elegido o se prepara para que se implemente por si solo de manera automática. Cabe destacar que en caso de que nuestro software sea una versión sustitutiva de un software anterior es recomendable valorar la convivencia de sendas aplicaciones durante un proceso de adaptación.

Mantenimiento.

consiste en la modificación del software después de la entrega al cliente para corregir fallos, mejorar el rendimiento o adaptar el producto a un entorno modificado. Existen cuatro tipos de mantenimiento.

- **Mantenimiento adaptativo.** Estará centrado en la modificación del producto según los cambios que se puedan producir tanto a nivel de software como de hardware. Es el mantenimiento más común.
- **Mantenimiento correctivo.** Se centrará en corregir los errores que puedan producirse una vez entregado el producto.
- **Mantenimiento perfectivo.** Perfeccionamiento de la aplicación tras el uso y descubrimiento de nuevas mejoras que pueden ser incluidas.
- **Mantenimiento preventivo.** Modificación del producto para mejorar la usabilidad, sin alterar sus especificaciones.

LENGUAJES DE PROGRAMACIÓN SEGÚN SU COMPILEACIÓN

La compilación es el proceso de traducir el código de un **lenguaje de alto nivel** al **lenguaje máquina** que leen las computadoras.

- COMPILADOS:** Se convierten en código binario a través de un compilador.
- INTERPRETADOS:** Requieren de un programa que lee la instrucción en **tiempo real** y la ejecuta.
- Escribes el código.**
- Pasa por un proceso de compilación que genera un archivo ejecutable.**
- Ese ejecutable es leído por la computadora.**

LENGUAJES DE PROGRAMACIÓN SEGÚN EL PARADIGMA

- FUNCIONALES:** En la programación hay **muchas maneras** de resolver un problema y cada una de ellas **constituye un paradigma**.
- MULTIPARADIGMA:** Pueden programar de varias maneras. El programador escoge el paradigma adecuado para cada trabajo.
- ORIENTADOS A OBJETOS:** Están orientados al **software**. Utilizan **menos instrucciones** para realizar una acción. Nos ayuda a entender **cómo funcionan las instrucciones** en la computadora. Te permite programar **aplicaciones y videojuegos**.
- REACTIVOS:** Permiten que los programas reaccionen a **eventos o cambios** en los datos. Existen librerías como RX que pueden **aplicarse a casi todos** los lenguajes.

LENGUAJES DE PROGRAMACIÓN ALTO NIVEL VS BAJO NIVEL

- ALTO NIVEL:** Es un lenguaje que **entiende los humanos**. Son **instrucciones para el procesador**.
- BAJO NIVEL:** Son **instrucciones para la máquina**.
- ASSEMBLY:** Puedes construir **sistemas operativos y núcleos**.

PROGRAMACION ORIENTADA A OBJETOS

¿QUÉ ES LA PROGRAMACIÓN ORIENTADA A OBJETOS?

- Los objetos se crean a partir de una **plantilla** llamada **clase**. Cada objeto es una **instancia** de su clase.
- En una aplicación los objetos están separados **pero se comunican** entre ellos. Usuarios, Profesores, Cursos, Ventas, Facturas.
- Puedes programar con este paradigma en la **mayoría** de lenguajes.
- Aprende a crear aplicaciones usando la POO en: ed.team/cursos/poo

¡PILARES DE LA PROGRAMACIÓN ORIENTADA A OBJETOS!

- ABSTRACCIÓN:** Es el proceso de definir los **atributos** y **métodos** de una clase.
 - ENCAPSULAMIENTO:** Protege la información de manipulaciones no autorizadas.
 - POLIMORFISMO:** Da la misma orden a varios objetos para que respondan de diferentes maneras.
 - HERENCIA:** Las clases hijo heredan **atributos** y **métodos** de las clases padre.
- Según el paradigma, la programación orientada objetos, se basa en estos 4 pilares. **Estos definen la simplicidad y la funcionalidad del código.**
- ¿Cómo utilizas la POO actualmente? ed.team/cursos/poo