UNIVERSIDAD NACIONAL GUILLERMO BROWN

Conceptos y Paradigmas de Lenguajes de Programación

UNIVERSIDAD NACIONAL GUILLERMO BROWN

Conceptos y Paradigmas de Lenguajes de PROGRAMACIÓN

Docente

Maximiliano Zorzoli.

Técnico en control y medición de procesos industriales
Analista de Sistemas
Ingeniero de Sistemas

Medios de comunicación

Clases virtuales en vivo:

Unirse a la reunión meet
Vínculo a la videollamada: https://meet.google.com/rts-stzr-zyj
link reunión :
Código de acceso: paradigmas
viernes 18,00 hs en vivo

Información de la cátedra:

CVG (Campus Virtual Global) de la Unab: https://campus.unab.edu.ar/?redirect=0

Pagina de la materia: https://campus.unab.edu.ar/paradigmas/

Classroom

Mails de contactos:

Profesor: Maximiliano Zorzoli– maxizorzoli@gmail.com

Consultas fuera de horario de clase: por mail, por CVG o por Classroom

Intro: Que significa un “paradigma” de lenguaje De programación?

Buscar en IA :

Se denominan Paradigmas de Programación a las formas de clasificar los lenguajes de
programación en función de sus características. Los idiomas se pueden clasificar en múltiples paradigmas. (wikipedia)

Algunos paradigmas se ocupan principalmente de las implicancias para el modelo de
ejecución del lenguaje, como permitir efectos secundarios o si la secuencia de operaciones está definida por el modelo de ejecución. Otros paradigmas se refieren principalmente a la forma en que se organiza el código, como agrupar un código en unidades junto con el estado que modifica el código. Sin embargo, otros se preocupan principalmente por el estilo de la sintaxis y la gramática.

Temas a Desarrollar en la materia

- Paradigmas de lenguajes de programación, imperativo, declarativa, orientado a objetos, funcional.
- Conceptos formales de sintaxis y semántica, métodos de definición
- Semántica operacional, nociones de semántica formal.
- Representación en ejecución.
- Elementos. Unidades recursivas.
- Unidades genéricas, alias y sobrecarga.
- Distintos sistemas de tipos.
- Encapsulamiento y abstracción de la información. Intérpretes y Compiladores

Lenguajes de Programación ? que son:

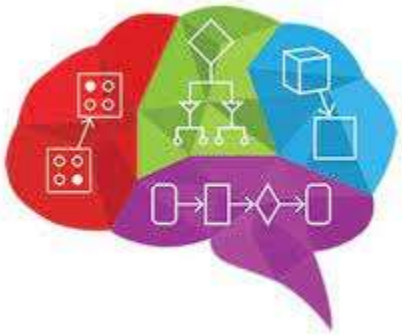
Los lenguajes de Programación son el corazón de la Ciencia de la Computación. Son herramientas que usamos para comunicarnos con las máquinas y también con las personas.

Cual es la idea ?

El valor de un lenguaje o de un concepto se debe juzgar según la forma en que **afecta la producción** de Software y a la facilidad con la que puede **integrarse** a otras herramientas" • Introducir, analizar y evaluar los conceptos más importantes de los lenguajes de programación.

Entendiendo los Paradigmas …

En esencia, los paradigmas de programación son como conjuntos de principios y directrices que determinan cómo pensamos y resolvemos problemas en el mundo de la programación.



Cada paradigma se basa en una filosofía única que influye en las estructuras de datos, los patrones de diseño y las técnicas de implementación que utilizamos. Algunos paradigmas se centran en la simplicidad y la legibilidad del código, mientras que otros priorizan la eficiencia y la concisión.

Que conseguiremos:

- ✓ Adquirir habilidad de apreciar y evaluar lenguajes, identificando los conceptos más importantes de cada uno de ellos y sus límites y posibilidades
- ✓ Habilidad para elegir, para diseñar, implementar o utilizar un lenguaje
- ✓ Enfatizar la abstracción como la mejor forma de manejar la complejidad de objetos y fenómenos

Si todo muy lindo... pero que es un código, o un paradigma?

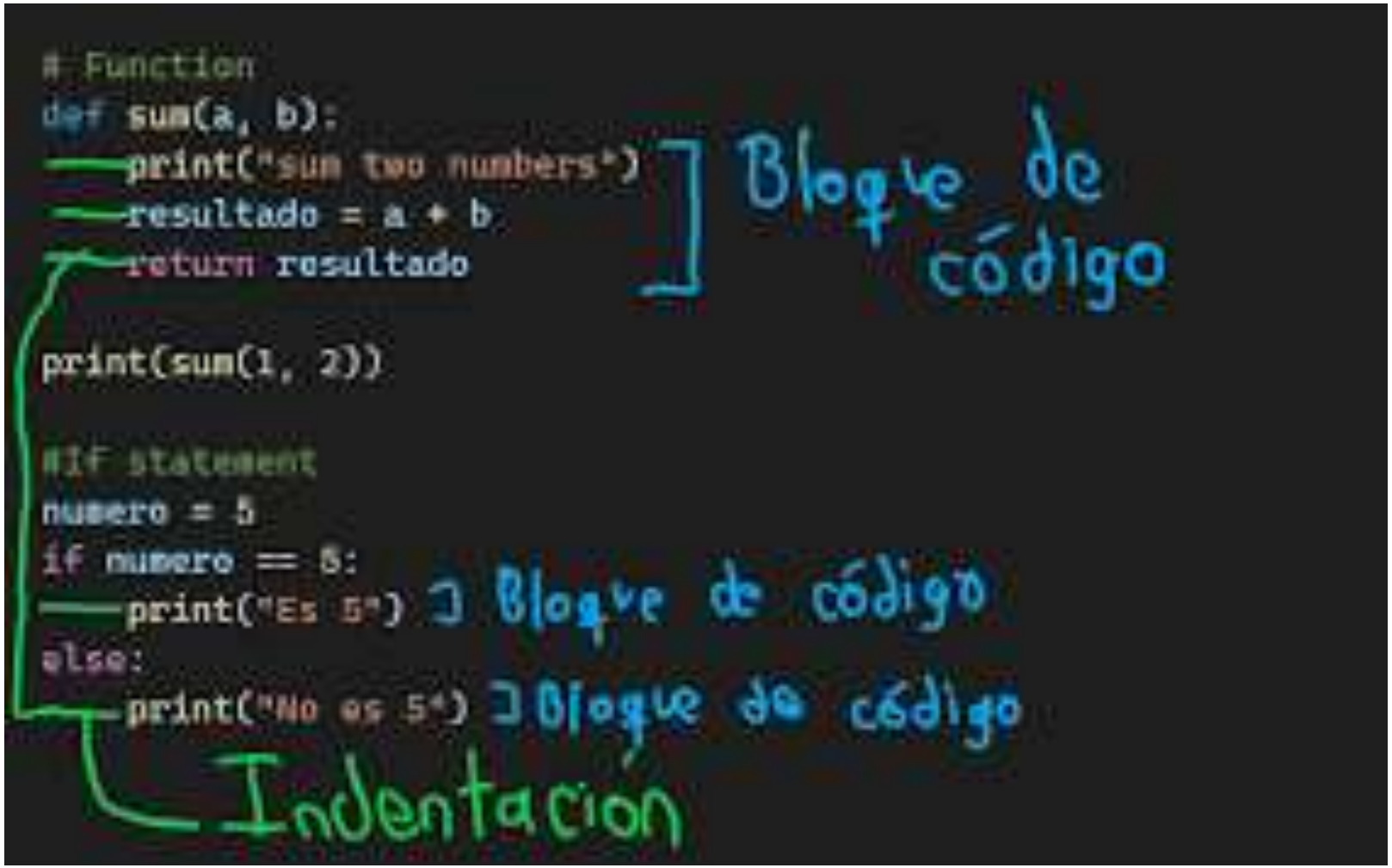
Como ya sabemos a esta altura ? Un código de programación es una secuencia de caracteres que le indica a una computadora qué acciones debe realizar. Se escribe en un lenguaje de programación específico que la computadora puede entender.

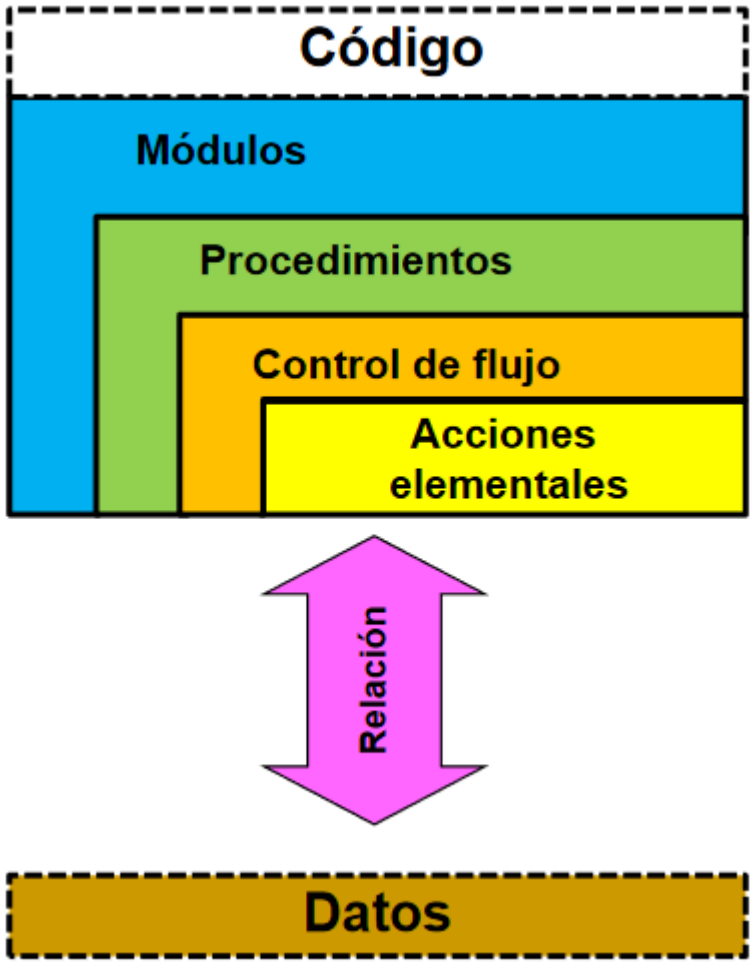
Cómo funciona la cosa

•Los códigos de programación se agrupan en líneas de código, que se separan por puntos y comas, saltos de línea u otros símbolos.

•Los lenguajes de programación actúan como un puente entre los humanos y las máquinas.

•Los lenguajes de programación permiten desarrollar programas, aplicaciones, sistemas operativos, páginas web y software


Bloque de código
Bloque de código
Bloque de código
Indentacion



Podemos definir la palabra **paradigma** como todo aquel **modelo, ejemplo o patrón** del que debemos guiarnos o debe seguirse en una determinada situación. En un sentido muy amplio de la palabra, podemos definir un paradigma como una **teoría o conjunto de teorías que sirve de modelo a seguir** para la resolución de problemas en diferentes ámbitos y circunstancias.

Bueno, seguimos entonces ;
Para qué estudiar conceptos de Lenguajes?

- ✓ Aumentar la capacidad para producir software.
- ✓ Mejorar el uso del lenguaje
- ✓ Elegir mejor un lenguaje
- ✓ Facilitar el aprendizaje de nuevos lenguajes
- ✓ Facilitar el diseño e implementación de lenguajes

Para poder evaluar los lenguajes , necesitamos criterios de evaluación, y por ende, definir algunos criterios deseables de diseño:

- Simplicidad y legibilidad
- Claridad en los bindings (enlaces, vinculaciones)
- Confiabilidad
- Soporte
- Abstracción
- Ortogonalidad
- Eficiencia

Entremos un poco mas en cada Criterio de evaluación definido anteriormente:

- **SIMPLICIDAD Y LEGIBILIDAD**

Los lenguajes de programación deberían:

– Poder producir programas fáciles de escribir y de leer.
– Resultar fáciles a la hora de aprenderlo o enseñarlo

Ejemplo de cuestiones que atentan contra esto:

– Muchas componentes elementales
– Conocer subconjuntos de componentes
– El mismo concepto semántico
– distinta sintaxis
– Distintos conceptos semánticos
- la misma notación sintáctica
– Abuso de operadores sobrecargados

Entremos un poco mas en cada Criterio de evaluación definido anteriormente:

- ## CLARIDAD EN LOS BINGINGS

Los elementos de los lenguajes de programación pueden ligarse a sus atributos o propiedades en diferentes momentos:

– Definición del lenguaje
– Implementación del lenguaje
– En escritura del programa
– Compilación
– Cargado del programa
– En ejecución

La ligadura en cualquier caso debe ser clara!!!

Entremos un poco mas en cada Criterio de evaluación definido anteriormente:

- **CONFIABILIADAD**

La confiabilidad está relacionada con la seguridad :

– Chequeo de tipos

• Cuanto antes se encuentren errores menos costoso resulta realizar los arreglos que se requieran. – Manejo de excepciones

• La habilidad para interceptar errores en tiempo de ejecución, tomar medidas correctivas y continuar.

Entremos un poco mas en cada Criterio de evaluación definido anteriormente:

- **SOPORTE**

Debería ser accesible para cualquiera que quiera usarlo o instalarlo

– Lo ideal sería que su compilador o intérprete sea de dominio público

• Debería poder ser implementado en diferentes plataformas

• Deberían existir diferentes medios para poder familiarizarse con el lenguaje: tutoriales, cursos textos, etc.

Entremos un poco mas en cada Criterio de evaluación definido anteriormente:

- **ABSTRACCIÓN**

Capacidad de definir y usar estructuras u operaciones complicadas de manera que sea posible ignorar muchos de los detalles.

– Abstracción de procesos y de datos

- **ORTOGONALIDAD**

Significa que un conjunto pequeño de constructores primitivos, puede ser combinado en número relativamente pequeño a la hora de construir estructuras de control y datos. Cada combinación es legal y con sentido.

significar independencia o desacoplamiento. En un sistema bien diseñado, el código de la base de datos será ortogonal al de la interfaz de usuario: se puede cambiar la interfaz sin afectar la base de datos y cambiar de bases de datos sin afectar la interfaz.

Si pensamos en un sistema puramente ortogonal, las operaciones no tienen efectos secundarios; cada acción cambia solo una cosa sin afectar a las otras (otro buen ejemplo es el monitor, al poseer controles ortogonales: al cambiar la luminosidad no afecta al contraste y viceversa).
El usuario comprende mejor si tiene un pequeño número de primitivas y un conjunto consistente de reglas de combinación.

Pascal y ADA NO son ortogonales, Rubyl SI
•En Pascal, por ejemplo: los proc y las funciones pueden ser pasadas por parámetro pero solo pueden pasar parámetros por valor, las funciones SOLO pueden devolver datos elementales, NO punteros, ni arreglos, etc. •En ADA, por ejemplo: los parámetros por valor pasados solo pueden ser elementales, por referencia: arreglos, etc.

- **ORTOGONALIDAD**

Digamos que tenés que manejar un helicóptero porque el piloto acaba de desmayarse ante ti, donde rápidamente identificas cuatro controles e ingenuamente piensas que cada uno controla sólo un aspecto del helicóptero.

Sin embargo la vida no es tan simple y descubres que cada control tiene un efecto secundario. Bajas la palanca izquierda pero también debes mover hacia atrás la palanca derecha y presionar el pedal derecho para estabilizar el helicóptero. En pocos segundos, tienes un sistema complejo donde cada cambio afecta a los demás controles.

Los controles de un helicóptero no son ortogonales

Entremos un poco mas en cada Criterio de evaluación definido anteriormente:

- **EFICIENCIA**

- Capacidad de lograr los resultados deseados con el mínimo posible de recursos.

➢ Tiempo y Espacio
➢ Esfuerzo humano
➢ Optimizable.

Como empezamos a aclarar un poco y separar las aguas: dos grandes grupos iniciales : Imperativo vs Declarativo

- La programación imperativa especifica los pasos que debe seguir la computadora para resolver un problema, mientras que la programación declarativa especifica el resultado deseado

	Programación imperativa	Programación declarativa
Enfoque	Instrucciones paso a paso	Describe el resultado final deseado
Código	Más voluminoso	Más liviano
Ventajas	Control preciso	Más legible y mantenible
Usos	Sistemas operativos, videojuegos	Bases de datos, interfaces web

Conceptos y Paradigmas de Lenguajes de PROGRAMACIÓN

La **programación imperativa** describe qué pasos hay que dar (How?) para obtener la solución de un problema La **programación declarativa** se enfoca en describe qué/cuál es (What?) la solución si entrar en los detalles su control de flujo.

Entremos un poco mas en cada Criterio de evaluación definido anteriormente:

- **Los Paradigmas de programación mas comunes son:**

Imperativo, en el que el programador instruye a la máquina cómo cambiar su estado, procedimental que agrupa las instrucciones en procedimientos

Orientado a objetos que agrupa las instrucciones con la parte del estado en el que operan, declarativo en el que el programador simplemente declara las propiedades del resultado deseado, pero no cómo calcularlo

Funcional en el que el resultado deseado se declara como el valor de una serie de aplicaciones de función,

Lógico en la que el resultado deseado se declara como la respuesta a una pregunta sobre un sistema de hechos y reglas,

Matemático en el que el resultado deseado se declara como la solución de un problema de optimización

Reactivo en el que se declara el resultado deseado con flujos de datos y la propagación del cambio

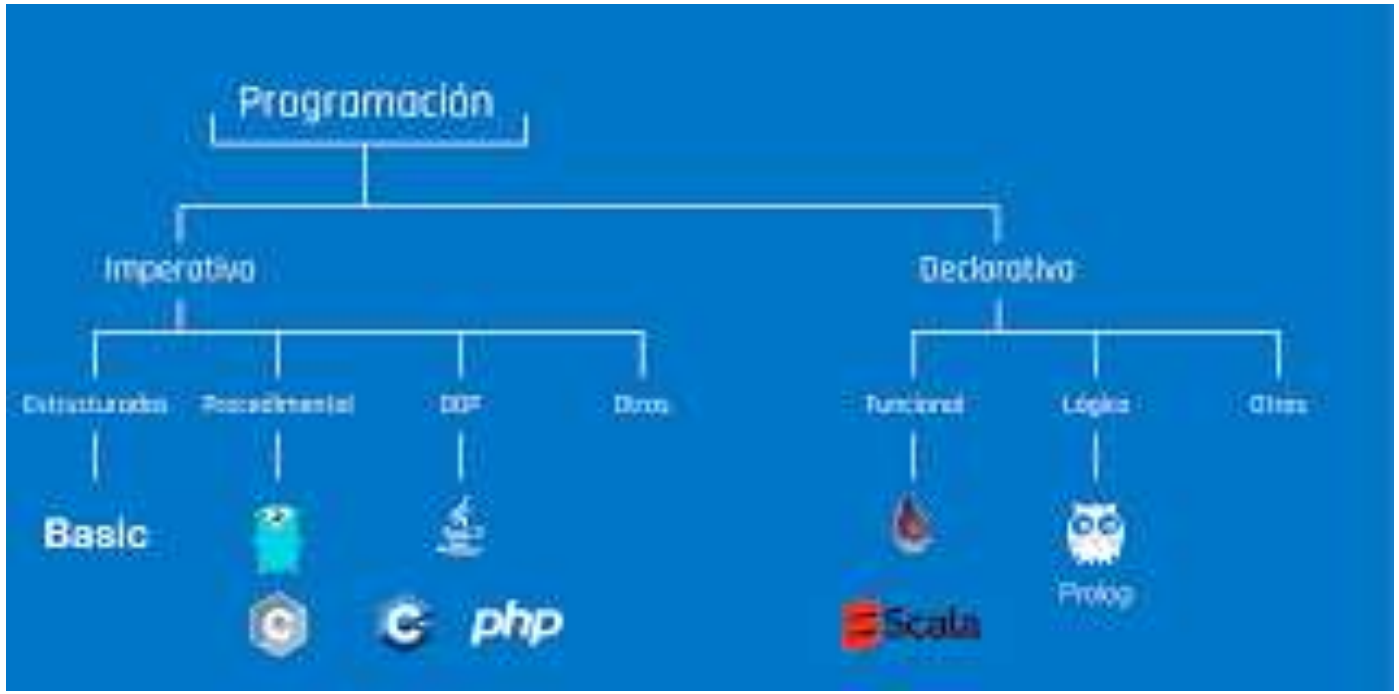
Declarativo, donde los enunciados constituyen una categoría de oraciones que apuntan a afirmar algo de manera clara y objetiva. Lo que se afirma puede ser un dato de la realidad circundante, una intención, un proyecto o un hecho. *Por ejemplo: Mañana es el cumpleaños de* mi madre.

Vamos de nuevo….

•**Programación Imperativa:** Se enfoca en describir paso a paso cómo realizar una tarea. Las instrucciones se ejecutan secuencialmente y se utilizan variables para almacenar y modificar estados.
•**Programación Declarativa:** En lugar de especificar cómo realizar una tarea, se describe qué resultado se desea obtener. Los lenguajes declarativos se centran en la lógica y las reglas, dejando al compilador o intérprete determinar cómo realizar la tarea.
•**Programación Orientada a Objetos (POO):** Se basa en el concepto de "objetos", que son instancias de clases que encapsulan datos y comportamientos relacionados. Los objetos interactúan entre sí mediante mensajes, lo que permite la reutilización y la organización del código de manera más estructurada.
•**Programación Funcional:** Se centra en funciones puras, que no tienen efectos secundarios y producen el mismo resultado para los mismos datos de entrada. La programación funcional evita el estado mutable y se basa en la composición de funciones para resolver problemas.
•**Programación Lógica:** Se basa en reglas lógicas y se centra en la inferencia y la resolución de problemas mediante consultas y hechos lógicos. Los lenguajes de programación lógica son útiles para resolver problemas complejos y aplicaciones de inteligencia artificial.
•**Programación Estructurada:** Es un enfoque que utiliza estructuras de control como bucles y condicionales para organizar el flujo de un programa de manera clara y comprensible.
•**Programación Modular:** Se enfoca en dividir un programa en módulos o unidades más pequeñas y autónomas. Esto facilita el mantenimiento, la reutilización y la colaboración en proyectos más grandes.
•**Programación Paralela/Concurrente:** Se refiere al diseño y ejecución de programas que realizan múltiples tareas al mismo tiempo, aprovechando la potencia de los sistemas con múltiples núcleos de procesador.
•**Programación Reactiva:** Se centra en la propagación automática de cambios y eventos dentro de un sistema, lo que permite reaccionar y responder a eventos en tiempo real.

UNIVERSIDAD NACIONAL GUILLERMO BROWN

Conceptos y Paradigmas de Lenguajes de PROGRAMACIÓN



Evolución y Aprendizaje Continuo

El mundo de la programación está en constante evolución, y los paradigmas de programación también evolucionan con él. Como desarrolladores, es esencial mantenerse actualizado con las últimas tendencias y enfoques para aprovechar al máximo las herramientas y técnicas disponibles.

- Video
- https://www.youtube.com/watch?v=xTLbG3_Rs_w
- https://www.youtube.com/watch?v=vM_2XUs7UGw

Conclusión

Los paradigmas de programación son esenciales para el desarrollo de software, ya que proporcionan pautas y estructura para abordar problemas de manera efectiva. Comprender los diferentes paradigmas y cómo aplicarlos de manera adecuada nos permite tomar decisiones informadas para crear código eficiente, mantenible y robusto. La elección del paradigma adecuado dependerá del contexto y los objetivos del proyecto, y estar dispuestos a aprender y adaptarse nos ayudará a enfrentar desafíos futuros en el emocionante mundo de la programación

- **Preguntas / dudas ???**