TEORÍA POO

***¿Qué es un lenguaje de programación?***

Un lenguaje de programación es un lenguaje formal que proporciona un programador, la capacidad y habilidad de escribir una serie de instrucciones o secuencias de órdenes en forma de algoritmos con el fin de controlar el comportamiento físico o lógico de un sistema informático

***¿Qué es el nivel de un lenguaje de programación?***

En resumen, el nivel de un lenguaje de programación indica cuánto se abstrae del hardware y qué tan fácil es para los humanos entender y escribir código en ese lenguaje.

Lenguaje de Bajo Nivel:

Estos lenguajes están más cerca del lenguaje máquina y dependen en gran medida de la arquitectura del hardware.

Ejemplos: Ensamblador (Assembly Language).

Ventajas: Ofrece un control muy detallado sobre los recursos del sistema, lo cual es ideal para la programación de sistemas embebidos o el desarrollo de sistemas operativos.

Lenguaje de Medio Nivel:

Combina características de los lenguajes de alto y bajo nivel.

Ventajas: Permite una manipulación directa de la memoria y el hardware mientras sigue ofreciendo abstracciones de alto nivel.

Desventajas: Aunque es más fácil que el lenguaje de bajo nivel, todavía puede ser más complicado que los lenguajes de alto nivel.

Lenguaje de Alto Nivel:

Cercano al lenguaje humano: Ofrecen una mayor abstracción y son más fáciles de usar para escribir y leer.

Ventajas: Más fácil de aprender, escribir y mantener, con una mayor portabilidad entre diferentes sistemas.

Desventajas: Menor control sobre el hardware y el rendimiento podría ser menos óptimo comparado con los lenguajes de bajo nivel.

***Lenguaje declarativo e imperativo***

Los lenguajes imperativos y declarativos son dos paradigmas diferentes en la programación, que reflejan distintas maneras de pensar y estructurar el código.

Lenguaje Imperativo

**Definición**: Un lenguaje imperativo se centra en **cómo** se debe realizar una tarea. El programador escribe una serie de instrucciones secuenciales que modifican el estado del programa.

**Analogía**: Es como dar una receta de cocina detallada, donde cada paso debe ser seguido en un orden específico.

Lenguaje Declarativo

**Definición**: Un lenguaje declarativo se centra en **qué** se quiere lograr sin especificar cómo hacerlo. El programador define el resultado deseado, y el lenguaje de programación maneja los detalles de cómo se logra ese resultado.

**Analogía**: Es como decirle a alguien "quiero una pizza", sin especificar cómo hacerla; el "cómo" es manejado automáticamente.

***Lenguaje de tipo dinamico y estatico:***  
Los lenguajes de programación pueden clasificarse según cómo manejan los tipos de datos en dos categorías principales: lenguajes de tipado dinámico y lenguajes de tipado estático. Esta clasificación se basa en cómo y cuándo se determinan los tipos de las variables en un programa.

Lenguaje de Tipado Dinámico

En un lenguaje de tipado dinámico, los tipos de las variables se determinan en **tiempo de ejecución**, es decir, mientras el programa está siendo ejecutado.

Lenguaje de Tipado Estático

En un lenguaje de tipado estático, los tipos de las variables se determinan en **tiempo de compilación**, es decir, antes de que el programa se ejecute.

***Tipado fuerte y debil***  
El tipado fuerte y débil se refieren a cómo un lenguaje de programación maneja las conversiones de tipos (también conocidas como coerciones) y las operaciones entre diferentes tipos de datos.

Tipado Fuerte

**Definición**: En un lenguaje de tipado fuerte, **las operaciones entre diferentes tipos de datos no se permiten sin conversión explícita**. Esto significa que el lenguaje es estricto en cuanto a cómo se manejan los tipos de datos y no permite mezclas inapropiadas sin que el programador haga una conversión clara y deliberada.

Tipado Débil

**Definición**: En un lenguaje de tipado débil, **las operaciones entre diferentes tipos de datos se permiten y a menudo se realizan automáticamente**. Esto significa que el lenguaje intentará realizar conversiones implícitas entre tipos cuando sea necesario para completar una operación.

***Paradigma:***

El **paradigma de un lenguaje de programación** es el enfoque o estilo fundamental que sigue el lenguaje para resolver problemas y estructurar el código. Los paradigmas de programación dictan cómo un programador piensa acerca de las soluciones y cómo organiza el código

C++ es un lenguaje de programación **multiparadigma**, lo que significa que soporta varios paradigmas de programación. Esto les da a los programadores una gran flexibilidad para elegir el enfoque que mejor se adapte a sus necesidades.

**TEORIA POO**

¿QUE ES? Es un paradigma de programación de objetos q interactúan entre sí para resolver problemas. Agrupa los datos y los procedimientos para manejarlos en una única entidad: el objeto

OBJETO: Un objeto es una unidad que engloba en sí mismo variables y funciones necesarios para el tratamiento de esos datos.

Tiene METODOS y ATRIBUTOS.

METODO: Función perteneciente a determinado objeto.

ATRIBUTO: Variable perteneciente a determinado objeto.

CLASE: Se puede considerar como un patrón para construir objetos.

HERENCIA: Capacidad de crear nuevas clases basándose en clases previamente definidas de las que se aprovechan ciertos datos y métodos, se desechan otros y se añaden nuevos.

JERARQUIA: Orden de subordinación de un sistema de clases.

POLIMORFISMO: Propiedad según la cual un mismo objeto puede considerarse como perteneciente a distintas clases.