**Adrián Tirado Ramos**

**Entorno De Desarrollo**

**1º De DAW**

Fundamentos de la poo

ÍNDICE

[1. Indica cuáles son sus fundamentos y los conceptos básicos 2](#_Toc164442506)

[Fundamentos 2](#_Toc164442507)

[Conceptos Básicos 2](#_Toc164442508)

[Clases y Objetos: 2](#_Toc164442509)

[Encapsulación: 2](#_Toc164442510)

[Herencia: 2](#_Toc164442511)

[Polimorfismo: 2](#_Toc164442512)

[Abstracción: 2](#_Toc164442513)

[2. En qué consiste el paradigma de la POO. 3](#_Toc164442514)

[Abstracción 3](#_Toc164442515)

[Encapsulación 3](#_Toc164442516)

[Herencia 3](#_Toc164442517)

[Polimorfismo 3](#_Toc164442518)

[3. Qué es OMG y UML 4](#_Toc164442519)

[OMG (Object Management Group): 4](#_Toc164442520)

[UML (Unified Modeling Language): 4](#_Toc164442521)

[Diagramas estructurales: 4](#_Toc164442522)

[Diagramas de comportamiento: 4](#_Toc164442523)

[Diagramas de interacción: 4](#_Toc164442524)

# Indica cuáles son sus fundamentos y los conceptos básicos

## Fundamentos

La Programación Orientada a Objetos (POO) es un paradigma de programación que se basa en la creación y manipulación de objetos, lo que facilita cosas como la reutilización, mantenimiento y organización del código.

## Conceptos Básicos

### Clases y Objetos:

**Clase**: Una clase es una plantilla que define cómo serán los objetos de un tipo específico, por ejemplo, una clase “Persona” podría tener atributos como “Nombre”, “Apellidos” y “Edad”, junto con comportamientos como “Hablar()” o “Caminar()”.

**Objeto**: Un objeto es una instancia de una clase. Representa una entidad con datos y comportamientos específicos. Por ejemplo, un objeto “Cliente” podría tener un nombre específico y realizar acciones como hacer una compra.

### Encapsulación:

La encapsulación implica ocultar los detalles internos de una clase y exponer solo lo necesario, los atributos y métodos privados se utilizan para lograr esto y así, los objetos interactúan con otros objetos a través de interfaces públicas.

### Herencia:

La herencia permite crear nuevas clases basadas en clases existentes.

Una clase derivada (subclase) hereda propiedades y comportamientos de su clase base (superclase), por ejemplo, una clase “Estudiante” podría heredar de la clase “Persona”.

### Polimorfismo:

El polimorfismo permite que los objetos de diferentes clases respondan de manera similar a un mismo método, por ejemplo, tanto un objeto “Perro” como un objeto “Gato” podrían tener un método “HacerSonido()”.

### Abstracción:

La abstracción se refiere a enfocarse en los aspectos esenciales de una entidad y ocultar los detalles innecesarios.

Las clases abstractas y las interfaces son herramientas para lograr la abstracción.

# En qué consiste el paradigma de la POO.

El paradigma de la Programación Orientada a Objetos (POO) es un enfoque fundamental en la construcción de software. A continuación, mencionaré sus pilares

## Abstracción

- La abstracción es la capacidad de representar conceptos complejos mediante modelos más simples y manejables.

- En POO, se crea una clase para representar un objeto del mundo real. Esta clase define sus propiedades (atributos) y comportamientos (métodos).

- Por ejemplo, si estamos modelando un sistema de biblioteca, podríamos tener una clase llamada `Libro` con atributos como título, autor y año de publicación.

## Encapsulación

- La encapsulación implica ocultar los detalles internos de una clase y exponer solo lo necesario para su uso externo.

- Se logra mediante el uso de métodos de acceso (como getters y setters) para interactuar con los atributos privados de una clase.

- Esto protege la integridad de los datos y facilita el mantenimiento y la evolución del código.

## Herencia

- La herencia permite crear nuevas clases basadas en clases existentes.

- Una clase derivada (o subclase) hereda atributos y métodos de su clase base (o superclase).

- Por ejemplo, si tenemos una clase Vehículo, podemos crear subclases como Automóvil y Motocicleta, que heredan características comunes de la clase base.

## Polimorfismo

- El polimorfismo permite que un objeto pueda presentarse de diferentes formas según el contexto en el que se utiliza.

- Se logra mediante la sobrecarga de métodos (mismo nombre, diferentes parámetros) y la sustitución de métodos (mismo nombre, diferentes implementaciones) en las subclases.

- Por ejemplo, un método calcularPrecio() puede comportarse de manera diferente para un Automóvil y una Motocicleta.

# Qué es OMG y UML

## OMG (Object Management Group):

OMG es un consorcio internacional sin fines de lucro que se dedica a promover y desarrollar estándares para sistemas orientados a objetos.

Fundado en 1989, OMG reúne a expertos de la industria, académicos y proveedores de tecnología para definir y estandarizar modelos y especificaciones.

Su objetivo es mejorar la interoperabilidad y la calidad del software mediante la adopción de estándares comunes.

Entre sus contribuciones más destacadas se encuentra el desarrollo y mantenimiento de UML.

## UML (Unified Modeling Language):

UML es un lenguaje estándar de modelado utilizado en ingeniería de software.

Proporciona una notación gráfica para visualizar, especificar, construir y documentar los artefactos de un sistema software.

Algunos aspectos clave de UML son:

### Diagramas estructurales:

* Diagrama de clases: Representa las clases, sus atributos, relaciones y métodos.
* Diagrama de objetos: Muestra instancias específicas de clases y sus relaciones.
* Diagrama de componentes: Ilustra la estructura de componentes y sus interacciones.

### Diagramas de comportamiento:

* Diagrama de secuencia: Describe la interacción entre objetos a lo largo del tiempo.
* Diagrama de actividades: Modela flujos de trabajo y procesos.
* Diagrama de estados: Representa los estados y transiciones de un objeto.

### Diagramas de interacción:

* Diagrama de colaboración: Muestra cómo los objetos colaboran en un escenario específico.
* Diagrama de comunicación: Enfoca en las interacciones entre objetos.

UML proporciona una herramienta visual poderosa para comunicar y diseñar sistemas de software complejos de manera clara y precisa.

En resumen, OMG y UML desempeñan un papel crucial en la estandarización y la mejora de la ingeniería de software, permitiendo una comunicación efectiva entre los desarrolladores y facilitando el diseño y la documentación de sistemas.